

AD-A038 891

NAVAL SEA SYSTEMS COMM/ND WASHINGTON D C

F/G 17/1

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUP--ETC(U)

DEC 75

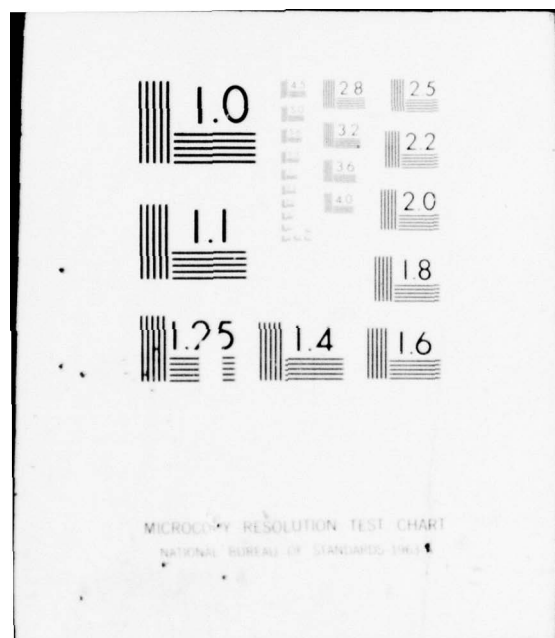
UNCLASSIFIED

NAVSEA-06H1/036-EVA/MOST-

NL

1 OF 2  
AD  
A038891







SEA 06H1/036-EVA/MOST-4

1 DECEMBER 1975

**NAVSEA OCEAN ENVIRONMENTAL  
ACOUSTIC DATA BANK**

**—NAVDAB—**

**IN SUPPORT OF  
MOBILE SONAR TECHNOLOGY DEVELOPMENT**

**VOLUME 3**



**NAVAL SEA SYSTEMS COMMAND  
DEPARTMENT OF THE NAVY  
WASHINGTON, D.C. 20362**

Approved for public release; distribution unlimited

**DISTRIBUTION STATEMENT A**

Approved for public release;  
Distribution Unlimited

AD A 038891

DDC FILE COPY

DDC

APR 28 1977

1  
SEA 06H1/036-EVA/MOST-4

1 DECEMBER 1975

NAVSEA OCEAN ENVIRONMENTAL  
ACOUSTIC DATA BANK

—NAVDAB—

IN SUPPORT OF  
MOBILE SONAR TECHNOLOGY DEVELOPMENT

VOLUME 3



NAVAL SEA SYSTEMS COMMAND  
DEPARTMENT OF THE NAVY  
WASHINGTON, D.C. 20362

Approved for public release; distribution unlimited.

REVISION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
DIST.	AVAIL. SEC. SPECIAL
A	

DDC  
RECEIVED  
APR 29 1977  
A

DECLASSIFICATION STATEMENT A

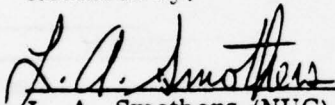
Approved for public release;  
Distribution Unlimited

## ADMINISTRATIVE STATEMENT

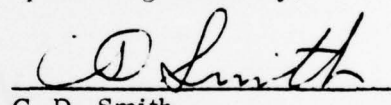
The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is a joint effort of the Naval Undersea Center (NUC), the Naval Underwater Systems Center (NUSC), the Naval Research Laboratory (NRL), and the Naval Oceanographic Office (NAVOCEANO), under the sponsorship of the Sonar Technology Office of the Naval Sea Systems Command (Task Area SF 52 552 601). Each of these organizations is represented in the NAVDAB Steering Group. The Steering Group has the responsibility for development of the data bank, which has been installed at NUC/San Diego, California, NUSC/New London, Connecticut, and NAVOCEANO/Washington, D.C.

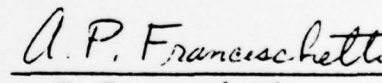
The NAVDAB Steering Group is grateful to G. F. Anderson of Computer Sciences Corporation and G. E. Miller of Arthur D. Little, Inc., for their assistance on this project.

Released by:

  
L. A. Smothers (NUC)  
NAVDAB Chairman

Sponsoring Authority:

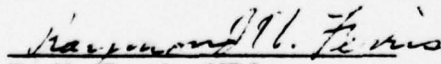
  
C. D. Smith  
NAVSEA

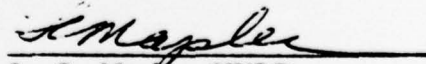
  
A. P. Franceschetti  
NAVSEA

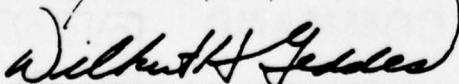
Steering Group

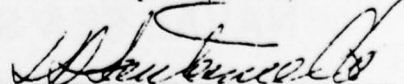
  
F. R. DiNapoli (NUSC)

  
K. V. Mackenzie (NAVOCEANO)

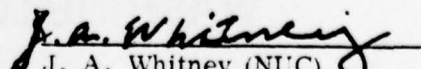
  
R. H. Ferris (NRL)

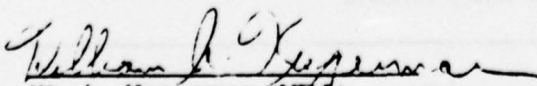
  
L. C. Maples (NUSC)

  
W. H. Geddes (NAVOCEANO)

  
S. R. Santaniello (NUSC)

  
R. F. Hosmer (NUC)

  
J. A. Whitney (NUC)

  
W. A. Kuperman (NRL)



SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER <b>NAVSEA-06H1/036-EVA/MOST-4</b>	2. GOVT ACCESSION NO.	3. RECIPIENT CATALOG NUMBER
4. TITLE (and Subtitle) <b>NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUPPORT OF MOBILE SONAR TECHNOLOGY DEVELOPMENT, VOLUME 3,</b>	5. TYPE OF REPORT & PERIOD COVERED <b>Final Rept.</b>	
6. AUTHOR(s) <b>NAVDAB Steering Group</b>	7. PERFORMING ORG. REPORT NUMBER <b>SF 52-552-601</b>	
8. PERFORMING ORGANIZATION NAME AND ADDRESS <b>Naval Undersea Center, Code 3073 San Diego, CA. 92132</b>	9. PROGRAM ELEMENT PROJECT, TASK AREA & WORK UNIT NUMBERS <b>Task No. 19324</b>	
10. CONTROLLING OFFICE NAME AND ADDRESS <b>Naval Sea Systems Command (NAVSEA 06H1-4) Washington, D.C. 20362</b>	11. REPORT DATE <b>1 Dec 1975</b>	
12. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) <b>(12) 129p.</b>	13. SECURITY CLASS. (of this report) <b>Unclassified</b>	
14. DISTRIBUTION STATEMENT (of this Report) <b>Approved for public release; distribution unlimited.</b>		
15. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) <b>(16) F52552 (17) SF52552601</b>		
16. SUPPLEMENTARY NOTES <b>Prepared in cooperation with the NAVDAB Steering Group.</b>		
17. KEY WORDS (Continue on reverse side if necessary; use block numbers) <b>Data bank Underwater Acoustics Environmental data Computer Programs</b>		
18. ABSTRACT (Continue on reverse side if necessary; use block numbers) <b>The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base of underwater acoustic and associated environmental data. This document provides detailed information on the creation phase computer programs.</b>		

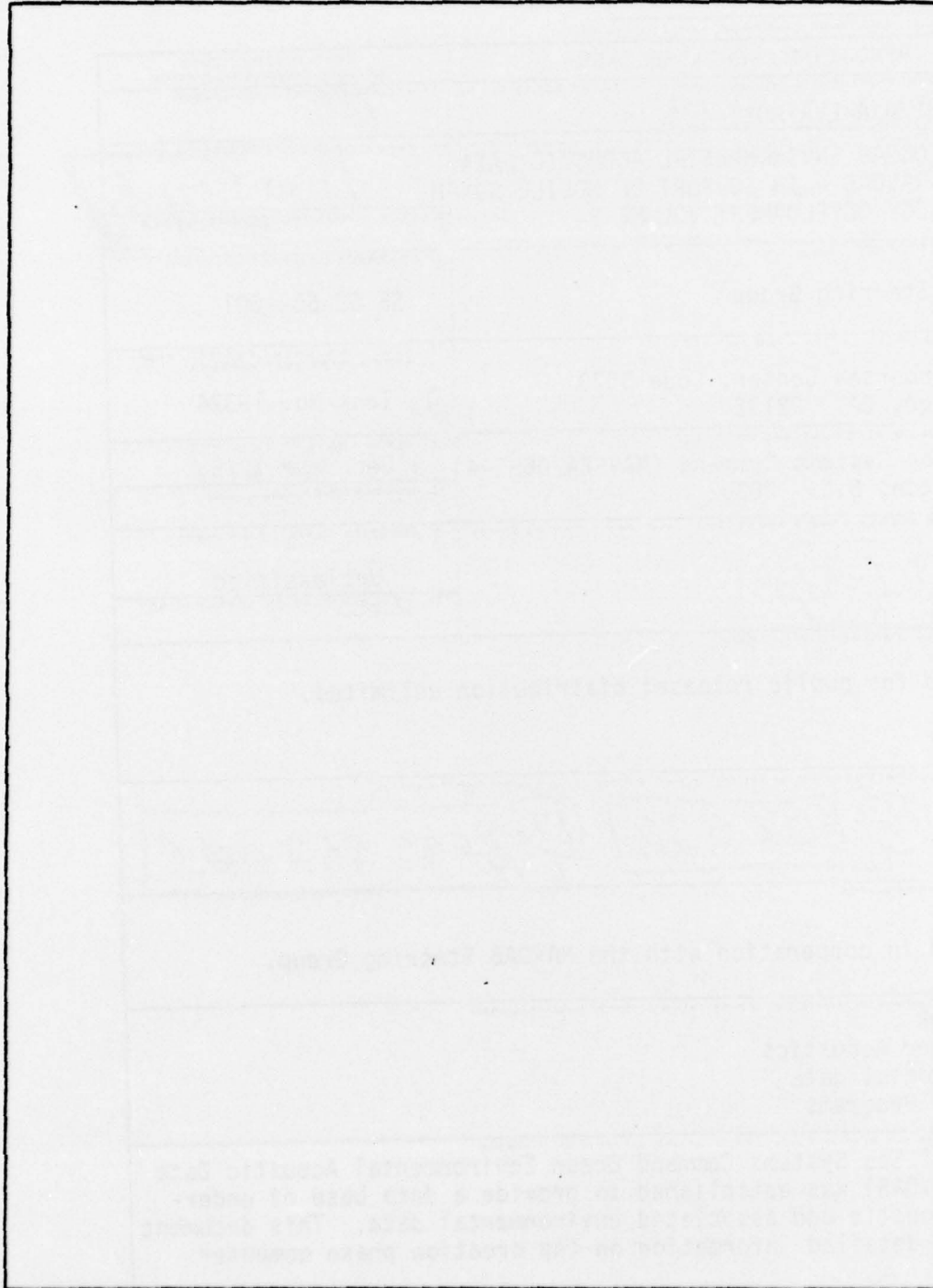
DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391345

4/B

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



DEPARTMENT OF THE NAVY  
NAVAL SEA SYSTEMS COMMAND  
WASHINGTON, D.C. 20362

FOREWORD

The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar application. NAVDAB is unique in that it is primarily for environmental and acoustic data taken concurrently and it is the only data bank of this type operated for mobile sonar application.

A set of five reports have been prepared to describe how NAVDAB works and how to use it. This is volume 3 of the set. The individual reports cover the following topics:

- Volume 1 User's Guide to Retrieval
- Volume 2 Input Format Guide
- Volume 3 Details of Creation Phase
- Volume 4 Details of Retrieval Phase
- Volume 5 Details of Miscellaneous Support Programs

Governmental and industrial activities desiring to submit or retrieve environmental acoustic data should contact:

Naval Undersea Center  
Code 3073  
San Diego, CA - 92132

Naval Underwater Systems Center  
Code TALA  
New London Laboratory  
New London, CT - 06320

Naval Oceanographic Office  
Code 3440  
Washington, D.C. - 20373

Copies of the five reports can be obtained through the Defense Documentation Center, Defense Supply Agency, Cameron Station, Alexandria, VA - 22314.

A handwritten signature in cursive script, reading "C.D. Smith", is located in the bottom right area of the page.

C.D. Smith, Director  
Sonar Technology Office,  
06H1/036

## CONTENTS

<u>Section</u>		<u>Page</u>
1.0	INTRODUCTION	1-1
2.0	DESCRIPTION OF THE DATA BASE	2-1
3.0	DESCRIPTION OF THE DATA ENTRY PROCESS	3-1

### Appendices

A.	Computer Program Listings	A-1
B.	Current NAVDAB Steering Group	B-1

## LIST OF ILLUSTRATIONS

### Figure

2-1	NAVDAB Data Base Structure	2-2
3-1 )		( 3-2
3-2 )	Flowcharts for Data Entry Phase	( 3-3
3-3 )		( 3-4
A-1	Data Entry Program Cross-Reference Map	A-4



## SECTION 1.0

### INTRODUCTION

The Naval Sea Systems Command (NAVSEA) Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar applications. The purpose of this document is to provide details of the computer programs which generate the NAVDAB data base.

This is Volume 3 of the five-volume set covering the NAVDAB computer program documentation. The individual volumes cover the following topics:

Volume 1. User's Guide to Retrieval

Volume 2. Input Format Guide

Volume 3. Details of Creation Phase

Volume 4. Details of Retrieval Phase

Volume 5. Details of Miscellaneous Support Programs

Copies of these documents may be obtained from the Defense Documentation Center.



## SECTION 2.0

### DESCRIPTION OF THE DATA BASE

Many factors affect the design of a data base and retrieval system. In the case of NAVDAB a large number and variety of data parameters were considered. To meet this requirement an open-ended tabular format, with variable record length, was selected to accommodate the initial sets of parameters and data and allow for expansion.

Another primary consideration in the data base design was the need for rapid access to all elements of all data. The data base must be suited to a versatile, efficient, random-access, retrieval system. Random-access retrieval techniques dictate, to a great extent, how data are structured on a storage device.

Other requirements of the NAVDAB system include: safeguard of classified information; allowance for operation in either batch or time-sharing mode; ease of updating and maintenance; and use of a universally-accepted computer language, as well as sufficient modularity, for ready adaptation to the different types of computers on which the system may be implemented.

In view of all these considerations, the data organization shown in Figure 2-1 was developed. It is a multilevel, hierarchically-organized, partially circular, linked-list structure, with large-capacity disk for primary storage. Four levels are involved:

1. **EXPERIMENT** — The highest level, pertains to the overall program of measurements — such as AMOS, FASOR I (Forward Area Sonar Research)
2. **CRUISE** — The second level is applicable to natural subsets of an **EXPERIMENT**, such as individual AMOS cruises or measurements taken within a specified area

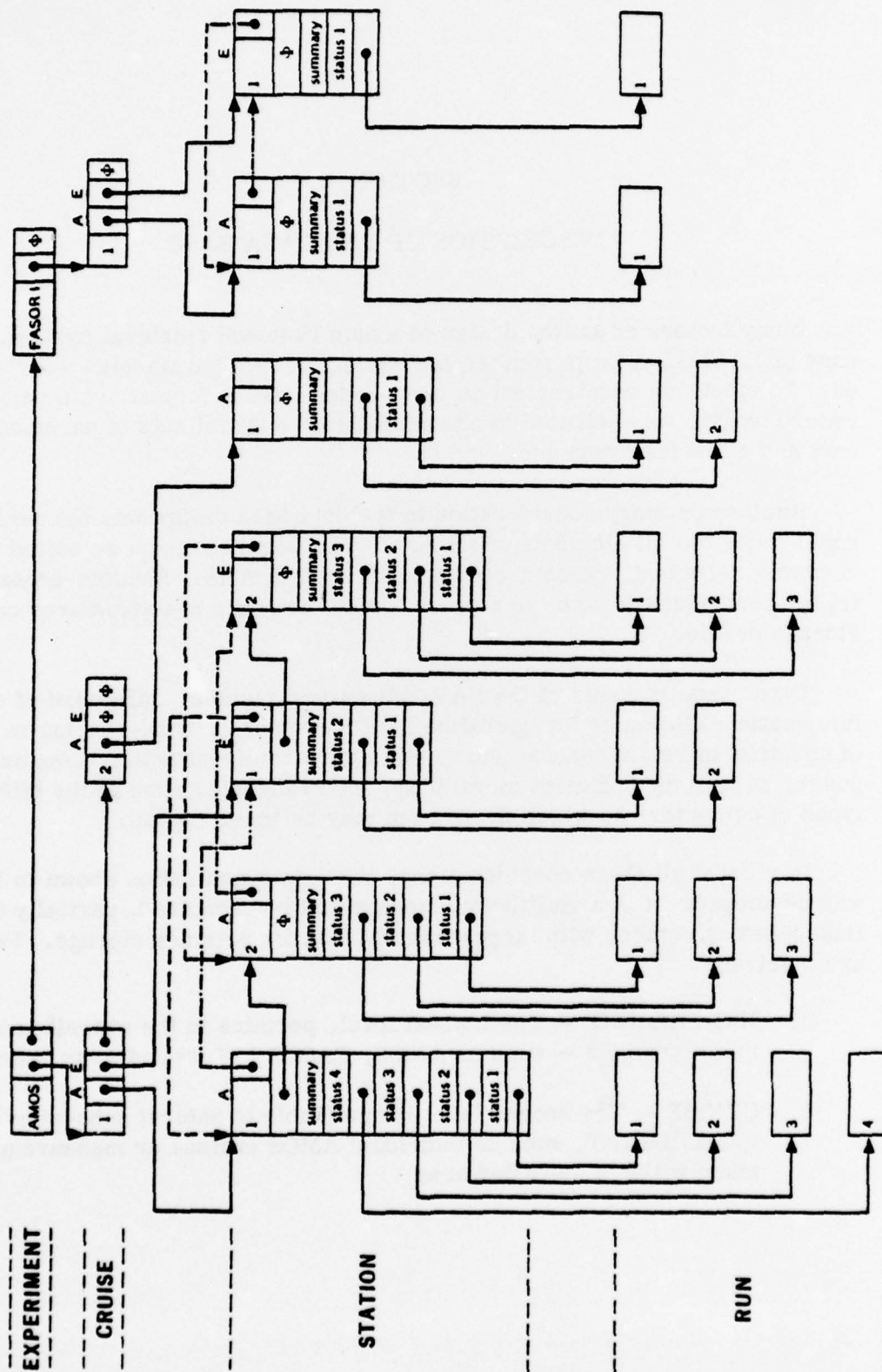


Figure 2-1. NAVDAB Data Base Structure.

3. STATION - A subdivision of a CRUISE: refers to measurements at or about a specific geographical location, such as FASOR II, Cruise 1, Station 1
4. RUN - The fourth and lowest level: designates a discrete set of measurements forming an element of a STATION, such as a propagation loss run or a hydrographic cast

Actual numerical data are stored only at the RUN level, which contains no other information. Key catalog information about the various Runs of a Station are stored at the STATION level, which contains, for all the Runs, chronological and geographical data and access category, and lists the acoustic and environmental parameters. The organization of the data is described for each Run, including the Run's storage location on the disk. Locations of associated Stations are also included. There are two types of Stations, acoustic and environmental, but the only environmental data currently approved by the Steering Group are those supporting accepted acoustic data.

Catalog information for the Stations of a Cruise is stored at the CRUISE level, which also contains storage locations of the first acoustic and environmental Stations and the next Cruise of the Experiment. In addition, there is an alphanumeric description of the features of the particular Cruise. These notes may be printed out in the retrieval process as background information.

The EXPERIMENT level contains summary information for the Experiment and directs flow down through the other levels. This flexible pointer structure allows additions or deletions of data without changing the programs that access the data.

## SECTION 3.0

### DESCRIPTION OF THE DATA ENTRY PROCESS

The program INIT0 is used to initialize the NAVDAB data base. The actual generation of the data base is accomplished by program MAIN. After examining the data base to determine the current status of its contents, MAIN begins reading card images which were prepared according to established NAVDAB input formats. (See Volume 2 of this documentation series.)

As each new Experiment, Cruise, or Station is read in, links (pointers) are established to associate it on the data base with the previous Experiment, Cruise, or Station. The schemes for processing constants, parameters, and variables are the same for environmental input card images as for acoustic card images. Numerous error checks are made as the data are processed.

Flowcharts for the data entry process are shown in Figure 3-1 through 3-3. Complete computer program listings are contained in Appendix A.

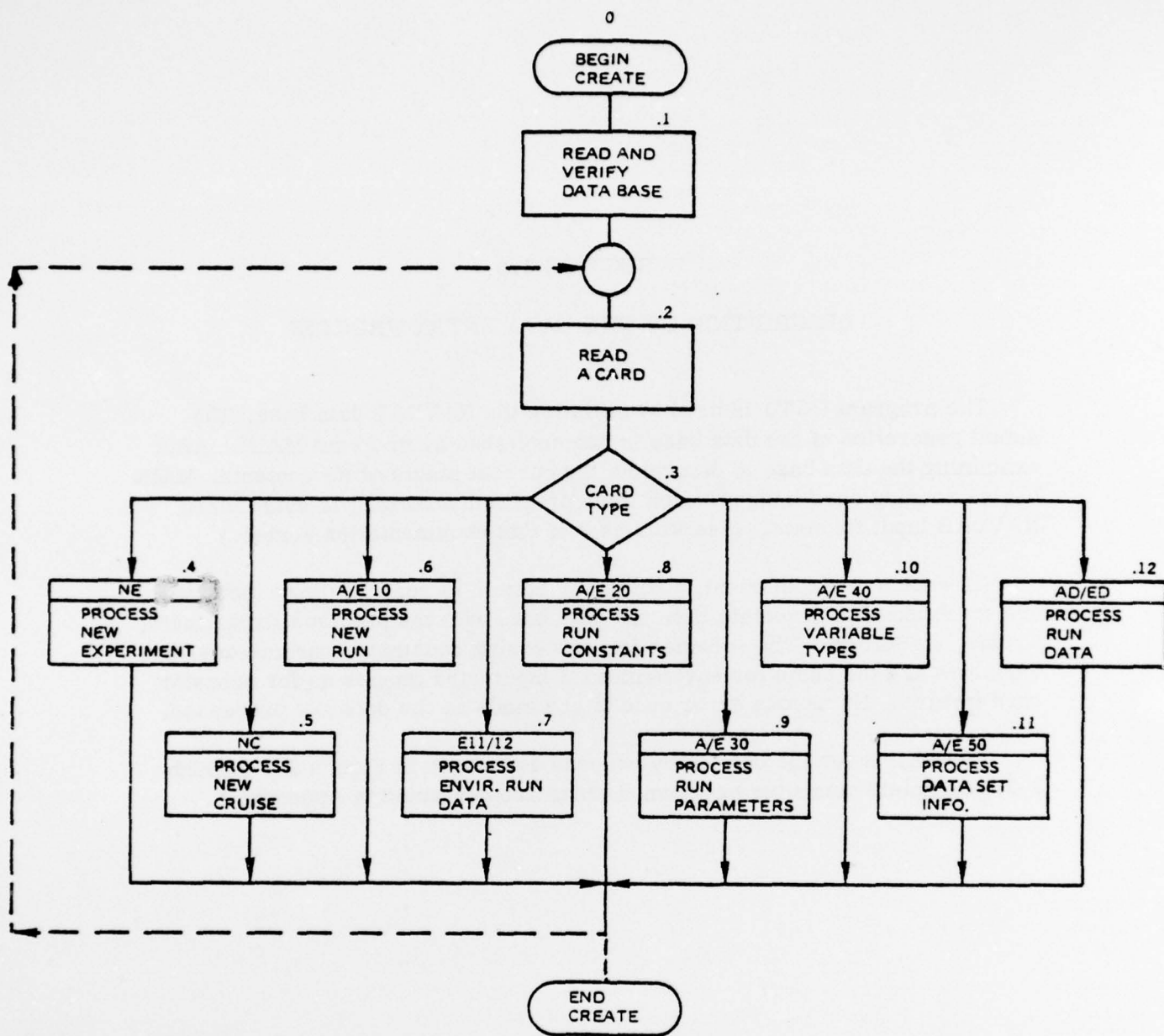


Figure 3-1.



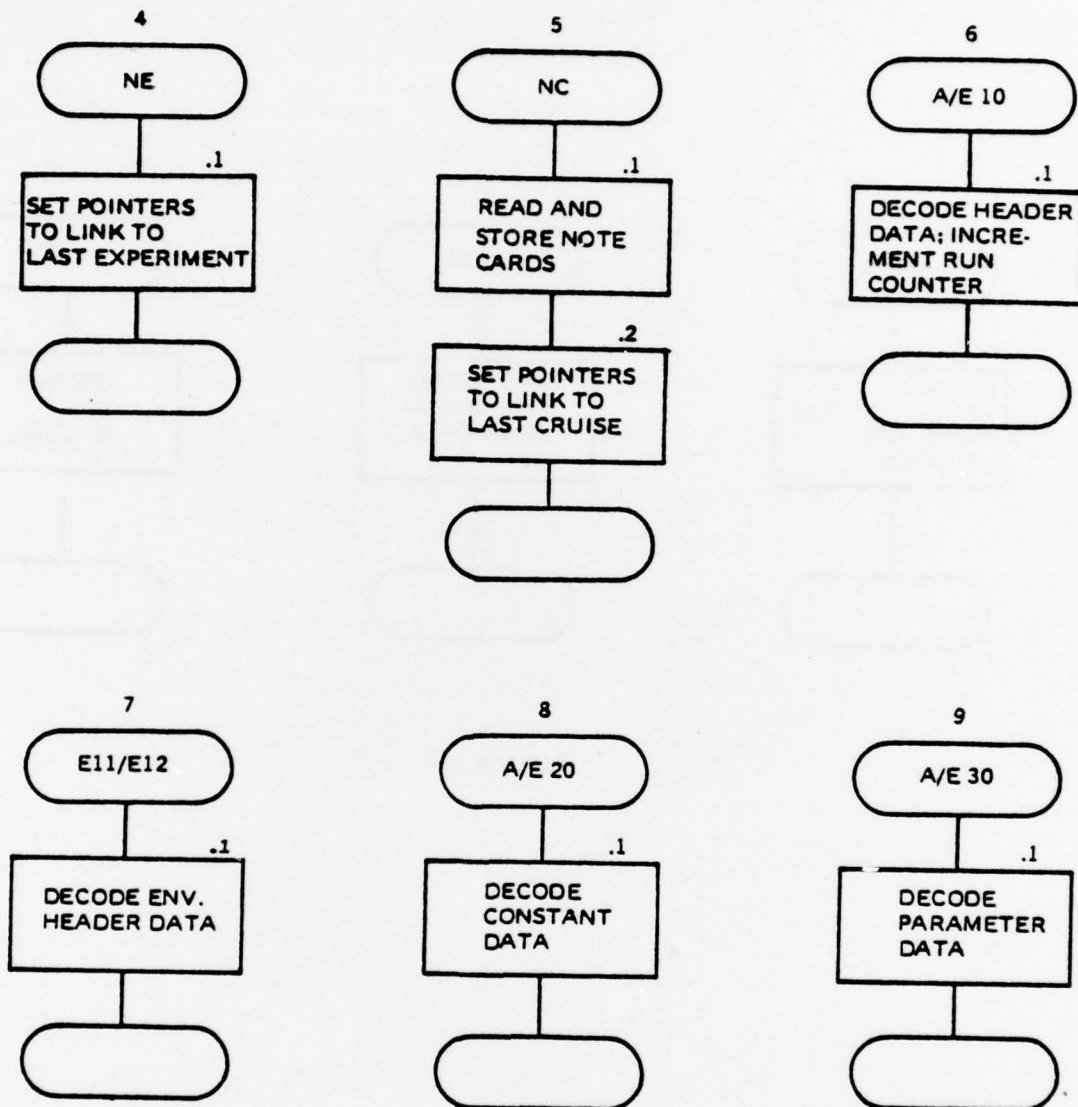


Figure 3-2.

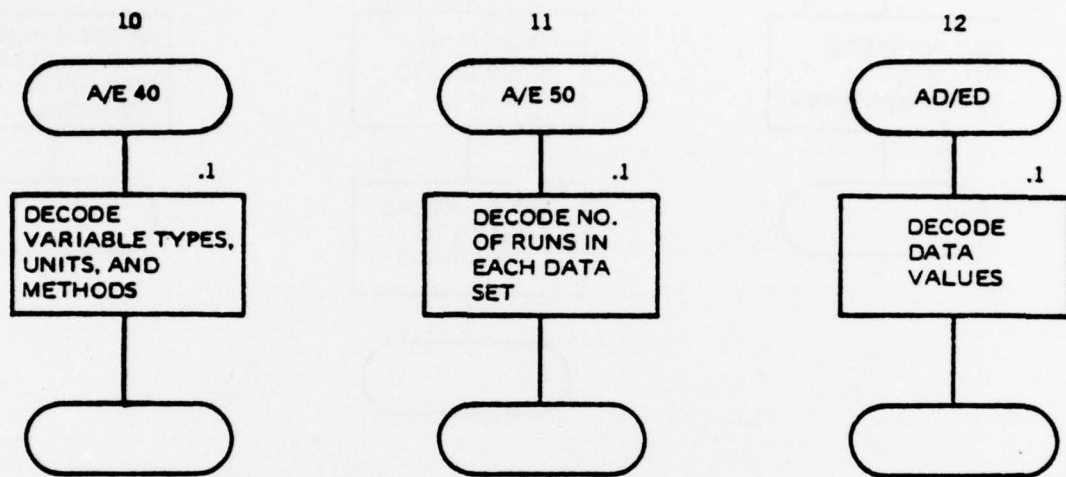


Figure 3-3.

APPENDIX A  
COMPUTER PROGRAM LISTINGS



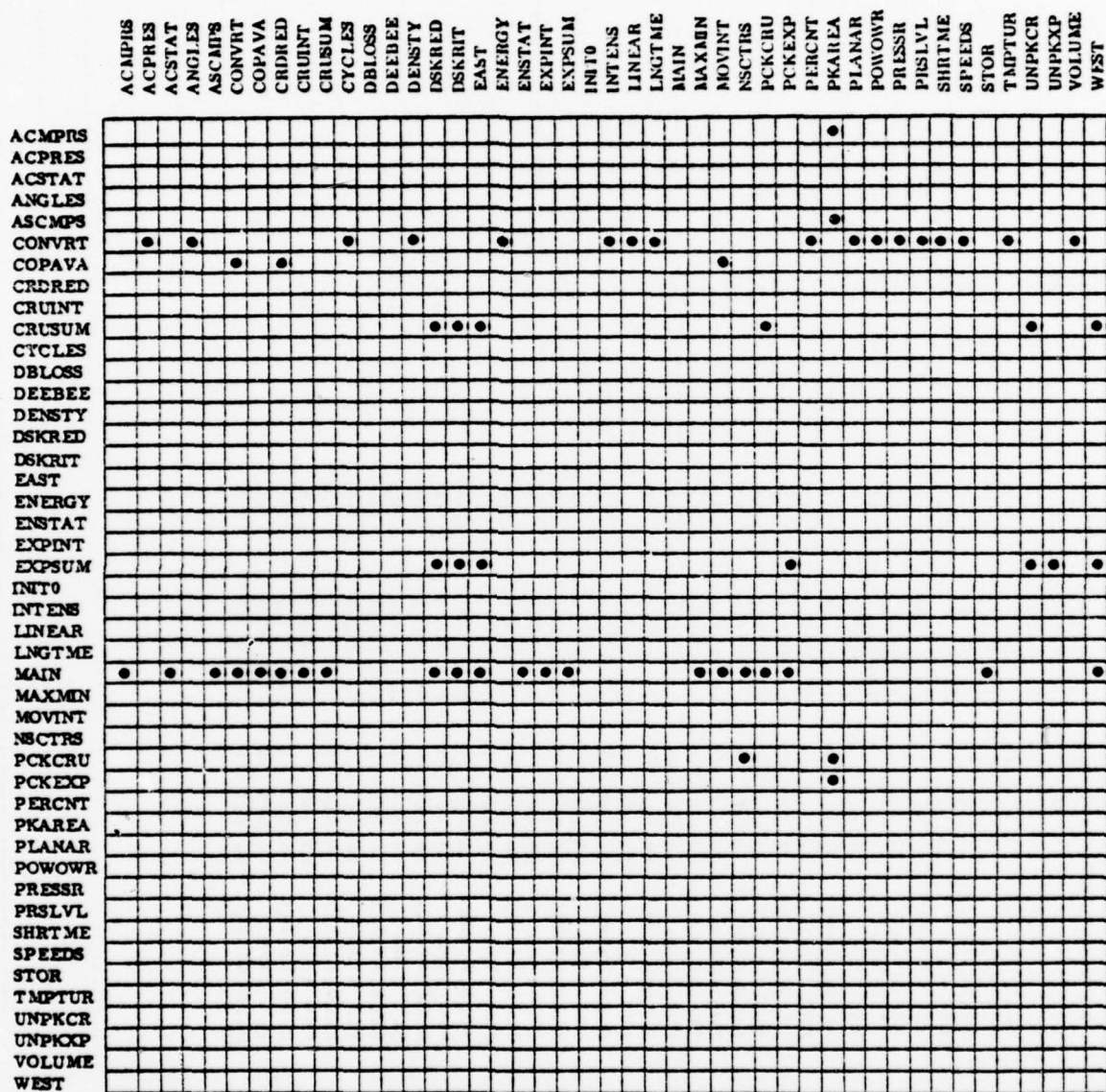
# ELEMENT TABLE

NAME	PAGE	TYPE	DATE	TIME	SEQ	SIZE-PRE,TEXT	(CYCLE WORD)	PSRMODE	LOCATION
ACHPRS	A-5	FOR SYMB	16 DEC 75	10:13:24	1	2	5	0	1792
ACHPRS	A-5	RELOCATABLE	16 DEC 75	16:09:24	2	2	5	0	1824
ACHPRS	A-9	FOR SYMB	22 AUG 74	16:23:46	3	12	5	0	1848
ACHPRS	A-11	RELOCATABLE	16 DEC 75	16:09:37	4	1	5	0	1940
ACHPRS	A-11	FOR SYMB	16 DEC 75	10:14:16	5	15	5	0	1949
ACHPRS	A-11	RELOCATABLE	16 DEC 75	16:09:49	6	1	5	0	1984
ACHPRS	A-13	FOR SYMB	01 NOV 74	12:02:12	7	13	5	0	1995
ACHPRS	A-13	RELOCATABLE	16 DEC 75	16:09:52	8	1	5	0	1908
ACHPRS	A-15	FOR SYMB	16 DEC 75	10:14:48	9	1	5	0	1919
ACHPRS	A-15	RELOCATABLE	16 DEC 75	16:10:04	10	1	5	0	1926
ACHPRS	A-16	FOR SYMB	31 OCT 75	15:10:50	11	27	5	0	1932
ACHPRS	A-16	RELOCATABLE	16 DEC 75	16:10:04	12	3	5	0	1959
ACHPRS	A-19	FOR SYMB	16 DEC 75	10:19:39	13	55	5	0	1987
ACHPRS	A-19	RELOCATABLE	16 DEC 75	16:10:25	14	2	44	44	2042
ACHPRS	A-25	FOR SYMB	28 MAY 75	20:43:35	15	4	5	0	2088
ACHPRS	A-25	RELOCATABLE	16 DEC 75	16:10:37	16	1	4	4	2092
ACHPRS	A-26	FOR SYMB	11 JUL 74	09:41:50	17	7	5	0	2097
ACHPRS	A-26	RELOCATABLE	16 DEC 75	16:10:37	18	1	2	2	2104
ACHPRS	A-27	FOR SYMB	16 DEC 75	10:14:41	19	20	5	0	2107
ACHPRS	A-27	RELOCATABLE	16 DEC 75	16:10:51	20	2	12	5	2127
ACHPRS	A-29	FOR SYMB	24 JUN 74	14:33:34	21	12	5	0	2141
ACHPRS	A-29	RELOCATABLE	16 DEC 75	16:10:51	22	1	6	6	2153
ACHPRS	A-31	FOR SYMB	06 SEP 74	09:23:35	23	15	5	0	2162
ACHPRS	A-31	RELOCATABLE	16 DEC 75	16:11:03	24	1	9	9	2177
ACHPRS	A-33	FOR SYMB	30 OCT 75	08:46:09	25	19	5	0	2187
ACHPRS	A-33	RELOCATABLE	16 DEC 75	16:11:01	26	1	15	15	2206
ACHPRS	A-35	FOR SYMB	28 JUN 74	13:34:17	27	12	5	0	2222
ACHPRS	A-35	RELOCATABLE	16 DEC 75	16:11:13	28	1	7	7	2234
ACHPRS	A-37	FOR SYMB	16 DEC 75	10:24:49	29	3	5	0	2242
ACHPRS	A-37	RELOCATABLE	16 DEC 75	16:11:24	30	1	4	4	2245
ACHPRS	A-38	FOR SYMB	16 DEC 75	10:25:04	31	3	5	0	2250
ACHPRS	A-38	RELOCATABLE	16 DEC 75	16:11:28	32	1	4	4	2253
ACHPRS	A-39	ELT SYMB	16 DEC 75	16:09:03	33	2	5	0	2258
ACHPRS	A-39	RELOCATABLE	16 DEC 75	15:50:52	34	1	3	3	2260
ACHPRS	A-40	FOR SYMB	24 JUN 74	14:33:40	35	13	5	0	2264
ACHPRS	A-40	RELOCATABLE	16 DEC 75	16:11:27	36	1	8	8	2277
ACHPRS	A-42	FOR SYMB	16 DEC 75	10:25:04	37	15	5	0	2286
ACHPRS	A-42	RELOCATABLE	16 DEC 75	16:11:38	38	1	10	10	2301
ACHPRS	A-44	FOR SYMB	16 DEC 75	10:25:15	39	5	5	0	2312
ACHPRS	A-44	RELOCATABLE	16 DEC 75	16:11:37	40	1	2	2	2317
ACHPRS	A-45	ELT SYMB	16 DEC 75	16:10:15	41	18	5	0	2320
ACHPRS	A-45	RELOCATABLE	16 DEC 75	15:56:48	42	2	11	11	2338
ACHPRS	A-47	FOR SYMB	16 DEC 75	10:25:52	43	6	5	0	2351
ACHPRS	A-47	RELOCATABLE	16 DEC 75	16:11:53	44	1	6	6	2357
ACHPRS	A-48	FOR SYMB	24 JUN 74	14:34:29	45	12	5	0	2364
ACHPRS	A-48	RELOCATABLE	16 DEC 75	16:12:00	46	1	7	7	2376
ACHPRS	A-50	FOR SYMB	30 AUG 74	13:34:51	47	15	5	0	2384
ACHPRS	A-50	RELOCATABLE	16 DEC 75	16:12:04	48	1	11	11	2399
ACHPRS	A-52	FOR SYMB	24 JUN 74	14:32:36	49	12	5	0	2411
ACHPRS	A-52	RELOCATABLE	16 DEC 75	16:12:03	50	1	8	8	2423
ACHPRS	A-54	FOR SYMB	16 DEC 75	10:26:28	51	276	5	0	2432
ACHPRS	A-54	RELOCATABLE	16 DEC 75	16:12:26	52	4	158	158	2708
ACHPRS	A-71	FOR SYMB	16 DEC 75	10:26:48	53	11	5	0	2870
ACHPRS	A-71	RELOCATABLE	16 DEC 75	16:12:25	54	1	6	6	2881

PAGE

NOVINT	A-78	ELT SYMB	16 DEC 75	16:09:13	55		11	5	0	1	2888
NOVINT	A-78	RELOCATABLE	16 DEC 75	15:52:49	56	1	4				2899
NSCTRS	A-79	FOR SYMB	16 DEC 75	15:26:28	57		4	5	0	1	2904
NSCTRS	A-79	RELOCATABLE	16 DEC 75	16:11:37	58	1	2				2910
PCKCRU	A-80	FOR SYMB	16 DEC 75	10:27:12	59		13	5	0	1	2913
PCKCRU	A-80	RELOCATABLE	16 DEC 75	16:11:36	60	1	8				2926
PCKEXP	A-82	FOR SYMB	16 DEC 75	10:27:14	61		9	5	0	1	2935
PCKEXP	A-82	RELOCATABLE	16 DEC 75	16:11:49	62	1	6				2944
PERCENT	A-83	FOR SYMB	17 JUL 75	13:52:12	63		12	5	0	1	2951
PERCENT	A-83	RELOCATABLE	16 DEC 75	16:13:52	64	1	8				2963
PRAREA	A-85	FOR SYMB	16 DEC 75	16:00:29	65		13	5	0	1	2972
PRAREA	A-85	RELOCATABLE	16 DEC 75	16:13:04	66	1	8				2985
PLANAR	A-87	FOR SYMB	03 SEP 75	09:15:53	67		13	5	0	1	2994
PLANAR	A-87	RELOCATABLE	16 DEC 75	16:13:14	68	1	9				3007
PO, QNR	A-89	FOR SYMB	24 JUN 74	14:34:03	69		12	5	0	1	3017
PO, QNR	A-89	RELOCATABLE	16 DEC 75	16:13:13	70	1	7				3029
PRESSR	A-91	FOR SYMB	22 AUG 74	10:25:08	71		14	5	0	1	3037
PRESSR	A-91	RELOCATABLE	16 DEC 75	16:13:26	72	1	10				3051
PRINTO	A-93	FOR SYMB	16 DEC 75	10:27:25	73		10	5	0	1	3062
PRINTO	A-93	RELOCATABLE	16 DEC 75	16:13:37	74	2	11				3072
PRSLVL	A-94	FOR SYMB	22 AUG 74	11:09:28	75		12	5	0	1	3085
PRSLVL	A-94	RELOCATABLE	16 DEC 75	16:13:39	76	1	7				3097
SHRTME	A-96	FOR SYMB	24 JUN 74	14:32:29	77		12	5	0	1	3105
SHRTME	A-96	RELOCATABLE	16 DEC 75	16:13:51	78	1	9				3117
SPFEDS	A-98	FOR SYMB	26 JUN 74	21:22:11	79		13	5	0	1	3127
SPFEDS	A-98	RELOCATABLE	16 DEC 75	16:13:49	80	1	10				3140
STOR	A-100	FOR SYMB	19 FEB 75	09:22:54	81		6	5	0	1	3151
STOR	A-100	RELOCATABLE	16 DEC 75	16:14:01	82	1	2				3157
THPTUR	A-101	FOR SYMB	17 JUL 75	13:45:33	83		13	5	0	1	3160
THPTUR	A-101	RELOCATABLE	16 DEC 75	16:14:02	84	1	11				3173
UNPKCR	A-103	ELT SYMB	16 DEC 75	16:09:14	85		10	5	0	1	3185
UNPKCR	A-103	RELOCATABLE	16 DEC 75	15:54:28	86	1	6				3195
UNPKXP	A-104	ELT SYMB	16 DEC 75	16:09:15	87		7	5	0	1	3202
UNPKXP	A-104	RELOCATABLE	16 DEC 75	15:54:27	88	1	5				3209
VOLUME	A-105	FOR SYMB	24 JUN 74	14:32:21	89		13	5	0	1	3215
VOLUME	A-105	RELOCATABLE	16 DEC 75	16:14:15	90	1	8				3228
WEST	A-107	ELT SYMB	16 DEC 75	16:09:16	91		2	5	0	1	3237
WEST	A-107	RELOCATABLE	16 DEC 75	15:55:02	92	1	3				3239
											3243

## CALLING ROUTINES



A-4



SELF, L

N,ACMPRS,,ACMPRS

SUBROUTINE ACMPRS (BFR,IPTR,LNTH)

IMPLICIT INTEGER (A-Z)

COMMON /RUN /PORA,MILAT,MINS,RELAT,RFNS,RILON,KELW,NFLON,KFEA,  
KING,KIDAY,KIYR,KITIM,RIZUP,KFMO,KFDAY,KFYR,KFITIM,  
RFZON,NAVCOJ,DTASRC,SYSSRC,SYSCRC,SNCTYP,RCNTYP,  
CLAS,DESCR,NAVDIR,NAVHT,MIUNIT,NAVPRD,PROUNT,  
SEASTA,ANDDIR,NDVEL,VELUNT,SALIN,SALHT,  
SHTUNT,SALPRD,SPDUNT,AEATHK,BUTDPH,UPHUNT,BTMSLP,  
BATHY,INFD(9),KSTAT(2),RUNNO  
REAL NAVDIR,NAVHT,NAVPRD,SEASTA,ANDDIR,NDVEL,SALDIR,  
SALHT,SALPRD,AEATHK,BUTDPH,BTMSLP

COMMON /CPV /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),  
NUATS,NUATS(5),NUATA,VALNG(2,30,50),DATA(3000)  
DIMENSION ICON(4,30),IPAR(30,30),IVLNG(2,30,50)  
EQUIVALENCE (CON(1,1),ICON(1,1)),IPAR(1,1),IPAR(1,1)),  
IVLNG(1,1),IVLNG(1,1))  
REAL CON,PAR,DATA,VALNG

DIMENSION BFR(1)  
REAL BFR

SET ISUB AS #GKING SUBSCRIPT. IPTR NOT CHANGED UNTIL RETURN

ISUB = IPTR

FLD(1,18,5FR(1502))=RUNNO

ISUB=ISUB + 1  
FLD ( 3,9,BFR(15JB) ) = NAVCOJ  
FLD ( 9,9,BFR(15UB) ) = DTASRC  
FLD (18,9,BFR(15UB) ) = CLAS  
FLD (27,9,BFR(15UB) ) = DESCR

ISUB = ISUB + 1

FLD ( 3,9,BFR(15UB) ) = SYSSRC  
FLD ( 9,9,BFR(15UB) ) = SYSCRC  
FLD (18,9,BFR(15UB) ) = SNCTYP  
FLD (27,9,BFR(15UB) ) = RCNTYP

COMPRESS AREA/TIME OF RUN

BEST AVAILABLE COPY

```

55. ISUB = ISUB + 1
56. CALL PKAREA (IPAR,MILAT,MILON,MILAY,RITIM,RIZON,
57. RELAT,MFLON,MFLAY,RETIM,REFZON,MFK(ISUB))

```

```

58. C
59. C
60. C

```

```

61. ISUB = ISUB + 4
62. FLD(1,36,MFK(ISUB)) = MSTAT(1)
63. FLD(2,36,MFK(ISUB+1)) = MSTAT(2)

```

```

64. C
65. C
66. C

```

```

67. INIT PTR TO KUM DATA
68. ISUB = ISUB + 2
69. MFK(ISUB) = 0

```

```

70. C
71. C
72. C
73. C

```

```

74. ISUB = ISUB + 1
75. FLD(1,9,MFK(ISUB)) = MCUN
76. FLD(1,9,MFK(ISUB)) = NPAR
77. FLD(18,9,MFK(ISUB)) = NPAR
78. FLD(27,9,MFK(ISUB)) = NPAR

```

```

79. C
80. C
81. C

```

```

82. IF (MCUN.EQ. 0) GO TO 45
83. DO 30 I=1,MCUN+2
84. ISUB = ISUB + 1
85. MFK(ISUB) = 0
86. FLD(1,9,MFK(ISUB)) = ICON(1,1)
87. FLD(18,9,MFK(ISUB)) = ICON(4,1)

```

```

88. J = J + 1
89. IF (J.GT. MCUN) GO TO 30
90. FLD(13,8,MFK(ISUB)) = ICON(1,J)
91. FLD(26,10,MFK(ISUB)) = ICON(4,J)

```

```

92. 30 CONTINUE

```

```

93. C
94. C

```

```

95. DO 40 I=1,MCUN
96. ISUB = ISUB + 1
97. MFK(ISUB) = CON(3,1)

```

```

98. 40 CONTINUE

```

```

99. C
100. C

```

```

101. C
102. C

```

```

103. 45 IF (IPAR.EQ. 0) GO TO 65
104. DO 60 I=1,IPAR

```

```

105. C
106. C

```

```

107. ISUB = ISUB + 1
108. FLD(1,9,MFK(ISUB)) = IPAR(1,1)
109. FLD(1,9,MFK(ISUB)) = IPAR(2,1)
110. FLD(17,10,MFK(ISUB)) = IPAR(4,1)

```

```

111. C

```

PACK NAME CODE & PROCESS CODE FOR ALL CONSTANTS

PACK ALL VALUES FOR PARAMETER(I)

112. C

113. C

JJ = IPAR(I,1)

DO 50 J=1,JJ

ISUB = ISUB + 1

BFR(ISUB) = PAR(4+J,1)

50 CONTINUE

118. C

119. C

60 CONTINUE

120. C

121. C

122. C

123. C

PACK NAME & PROCESS CODES FOR ALL VARIABLES

65 IF (NVAR - 64) GO TO 101

DO 75 I=1,NVAR,2

ISUB = ISUB + 1

BFR(ISUB) = C

FLD(3,6,BFR(ISUB)) = IVAR(1,1)

FLD(8,10,BFR(ISUB)) = IVAR(3,1)

J = 1 + 1

IF (J - 64) NVAR) GO TO 70

FLD(18,8,BFR(ISUB)) = IVAR(1,J)

FLD(26,10,BFR(ISUB)) = IVAR(3,J)

70 CONTINUE

125. C

126. C

127. C

PACK KNOWS IN EACH DATA SET INTO BFR.

DO 75 I=1,NDATS,2

ISUB = ISUB + 1

BFR(ISUB) = 0

FLD(5,18,BFR(ISUB)) = NROWS(I)

J = 1 + 1

IF (J - 64) NDATS) GO TO 75

FLD(18,14,BFR(ISUB)) = NROWS(J)

75 CONTINUE

130. C

131. C

132. C

DO 100 K=1,NDATS

DO 90 J=1,NVAR

135. C

136. C

137. C

DO 80 I=1,2

ISUB = ISUB + 1

BFR(ISUB) = VALRNG(I,J,K)

80 CONTINUE

140. C

141. C

142. C

90 CONTINUE

100 CONTINUE

145. C

146. C

147. C

COMPUTE LENGTH

101 LETH = ISUB + 1 - IPTH

150. C

151. C

152. C

FLD(18,14,BFR(IPTH)) = ISUB + 1

IPTH = IPTH + 1

155. C

156. C

BEST AVAILABLE COPY

ALL DONE

C

167.  
170.  
171.  
172.

RETURN  
END



```

1. SUBROUTINE ACPRES(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ACPRES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,3)/48,49,50/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,J),1=2,3),J=1,3)/'MICRO ','BARS ','
18. C 'PASCAL','S ','MICRO ','PAS '
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='MICRO '
23. C UNITS(4)='PASCAL'
24. C
25. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF (UNICODE.EQ.0) UNICODE=-50
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1,1=1,3)/ 1.05,1.03,10.J/
32. C
33. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
34. C CODE INTO 'UNITS'.
35. C
36. C ICODE=IABS(UNICODE)
37. C DO 2 1=1,3
38. C IF (ICODE.NE.ENTRY(1,I)) GO TO 2
39. C IF (UNICODE.EQ.0) VALUE=VALUE*FACTOR(I)
40. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
41. C GO 1 J=1,2
42. C UNITS(J)=ENTRY(J+1,1)
43. C 1 CONTINUE
44. C
45. C AT THIS POINT, CONVERSION IS COMPLETE.
46. C
47. C RETURN
48. C
49. C ELSE, CHECK REST OF TABLE.
50. C
51. C 2 CONTINUE
52. C
53. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
54. C

```



```

55.  UNITS(1)=UNITS ,
56.  UNITS(2)=ERROR ,
57.  PRINT IUI,ICODE
58.  IUI FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ACOUSTIC PRESSURE TABLE,')
59.  RETURN
60.  END

```

## SUBROUTINE ACSTAT (STATUS)

1. C  
2. C  
3. C  
4. C  
5. C  
6. C  
7. C  
8. C  
9. C  
10. C  
11. C  
12. C  
13. C  
14. C  
15. C  
16. C  
17. C  
18. C  
19. C  
20. C  
21. C  
22. C  
23. C  
24. C  
25. C  
26. C  
27. C  
28. C  
29. C  
30. C  
31. C  
32. C  
33. C  
34. C  
35. C  
36. C  
37. C  
38. C  
39. C  
40. C  
41. C  
42. C  
43. C  
44. C  
45. C  
46. C  
47. C  
48. C  
49. C  
50. C  
51. C  
52. C  
53. C  
54. C

COMMON /CPV /ICOD,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),  
MDATS,MK045(50),NDATA,VALNG(2,30,50),DATA(30000)

• DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)

• EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),  
(VALNG(1,1),IVLRNG(1,1))

• REAL CON,PAR,DATA,VALNG

DIMENSION STATUS(2)

CLEAR STATUS

STATUS(1) = 0

STATUS(2) = 0

CHECK CONSTANTS

IF (NCON .EQ. 0) GO TO 50

PROCESS CONSTANTS

DO 40 I=1,NCON

ICOD = IABS(ICON(I,1) - 1)

ISUB = 1

IF (ICOD .LE. 35) GO TO 10

ISUB = 2

ICOD = IABS(ICOD - 36)

IF (ICOD .LT. 35) GO TO 10

WRITE (6,4) ICON(I,1)

4 FORMAT (1 ENROR -- ACSTAT - ILLEGAL CODE\*,110)

STOP

10 FLD(ICOD,1,STATUS(ISUB)) = 1

40 CONTINUE

CHECK PARAMETERS

50 IF (NPAR .EQ. 0) GO TO 70

PROCESS PARAMETERS

GO 65 I=1,NPAR

ICOD = IABS(IPAR(2,1) - 1)

BEST AVAILABLE COPY

```

55. C
56. ISUB = 1
57. IF (ICOD .LE. 35) GO TO 60
58. ISUB = 2
59. ICD = IABS(ICOD - 36)
60. C
61. IF (ICOD .LT. 35) GO TO 60
62. WRITE (6,4) IPAR(1,1)
63. STOP
64. C
65. FLD(ICOD,1,STATUS,ISUB) = 1
66. 65 CONTINUE
67. C
68. CHECK VARIABLES
69. C
70. 72 IF INVAR .EQ. 0) GO TO 100
71. C
72. PROCESS VARIABLES
73. C
74. 00 90 I=1,NVAR
75. C
76. ICD = IABS(IVAR(1,1) - 1)
77. C
78. ISUB = 1
79. IF (ICOD .LE. 35) GO TO 80
80. ISUB = 2
81. ICD = IABS(ICOD - 36)
82. C
83. IF (ICOD .LT. 35) GO TO 80
84. WRITE (6,4) IVAR(1,1)
85. STOP
86. C
87. FLD(ICOD,1,STATUS,ISUB) = 1
88. 90 CONTINUE
89. C
90. 100 RETURN
91. END

```

2FOR,SA H,ANGLES,,ANGLES

```

1. SUBROUTINE ANGLES(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ANGLES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,3)/37,38,85/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1=2,3)/2=1,3)/.RADIANS,S
18. C .DEGREE,S
19. C .FTMMS/1,INTCLML/
20. C
21. C SET CONVERSION FACTORS.
22. C
23. C DATA (FACTOR(1),1=1,2)/1.000,.317453291700/
24. C
25. C ENTER INTERNAL DEGREE UNITS.
26. C
27. C UNITS(3)=.DEGREE,
28. C UNITS(4)=S
29. C
30. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
31. C
32. C IF (UNICODE.EQ.0) UNICODE=-37
33. C
34. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
35. C CODE INTO 'UNITS'.
36. C
37. C ICODE=ABS(UNICODE)
38. C DO 2 I=1,2
39. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
40. C IF (UNICODE.GE.0) VALUE=VALUE*FACTOR(1)
41. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(1)
42. C DO 1 J=1,2
43. C UNITS(J)=ENTRY(J,1)
44. C 1 CONTINUE
45. C
46. C AT THIS POINT, CONVERSION IS COMPLETE.
47. C
48. C RETURN
49. C
50. C ELSE, CHECK REST OF TABLE.
51. C
52. C 2 CONTINUE
53. C IF (ICODE.NE.ENTRY(1,1)) GO TO 3
54. C IF (UNICODE.GE.0) VALUE=ATAN(VALUE/1000.17,0174533)
55. C IF (UNICODE.LT.0) VALUE=ATAN(VALUE*.3174533)*1000.

```

```

55.  UNITS(1)=ENTRY(2,3)
56.  UNITS(2)=ENTRY(3,3)
57.  RETURN
58.  C
59.  C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
60.  3 UNITS(1)=UNITS *
61.  UNITS(2)=ERROR *
62.  PRINT IC,ICODE
63.  101 FORMAT(' ERROR--UNIT CODE ',I3,' NOT IN ANGLES TABLE. ')
64.  RETURN
65.  END

```



1. SUBROUTINE ASCHPS(STAUFN,TYPE)  
2. IMPLICIT INTEGER(A-Z)

3. C  
4. C  
5. C

6. COMMON /ASTAIN/SANO,SANKUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,  
7. SAFDAT,SASTAT(2),PNATAS,PENVS,SAIFLG

8. C  
9. C  
10. C

11. DIMENSION STAUFN(1)

12. CALL OUT STAUFN(2-10)

13. C  
14. C  
15. C

16. DO I=2,10  
17. STAUFN(I)=0

18. C  
19. C  
20. C

21. FLD(15,3,STAUFN(1))=TYPE  
22. FLD(15,18,STAUFN(2))=SANO  
23. FLD(18,18,STAUFN(2))=SANKUN

24. C  
25. C  
26. C

27. CALL PKAREA( ,SAILAT,SAILON,SAIDAT,D, , ,  
28. SAFLAT,SAFLON,SAFDAT,U, , ,STAUFN(J))

29. C  
30. C  
31. C

32. STAUFN(7)=SASTAT(1)  
33. STAUFN(8)=SASTAT(2)

34. C  
35. C  
36. C

37. STAUFN(9)=PNATAS  
38. RETURN  
39. END

40. C  
41. C  
42. C

43. PACK POINTED TO RELEVANT ENVIRONMENTAL (ACOUSTIC) STATION.

44. C  
45. C  
46. C

47. C  
48. C  
49. C

50. C  
51. C  
52. C

53. C  
54. C  
55. C

56. C  
57. C  
58. C

59. C  
60. C  
61. C

OFOR,SW N.CONVART,CONVRT

```

1. SUBROUTINE CONVERT(VALUE,INDKS,UNICDE,UNITS)
2. INTEGER TABLE,UNICDE
3. C
4. C SET UP TABLE TO MAP VARIABLE CODE TO PROPER TABLE NUMBER.
5. C
6. DIMENSION TABLE(144),UNITS(4)
7. DATA NULL /05763777777/
8. DATA (TABLE(I),I=1,72) /00401,00401,07712,07712,00018,02204,02204,00401,
9. 00401,00018,00401,00401,00401,00018,00501,00018,
10. 03808,07712,07712,00018,00401,02204,00018,
11. 00401,00401,00018,00018,02204,00000,03808,
12. 07712,03808,03208,02908,03808,03808,
13. 3800/
14. C
15. DATA TABLE(I),I=73,144) /00000,00401,02204,00000,03808,03204,00000,
16. 00401,02204,00000,00000,00000,00401,00401,
17. 00401,03807,04309,02908,00600,00301,00301,
18. 04317,04209,04209,04209,04209,04209,00401,00501,
19. 1400/
20. C
21. C
22. C UNITS(1)=UNITLE
23. UNITS(2)=55
24. UNITS(3)=UNITLE
25. UNITS(4)=55
26. C
27. C
28. C IF (FLD(0,36,VALUE).EQ.FLD(0,36,ANULL)) RETURN
29. C
30. C
31. C
32. C MAP CODES 1-72 AND 131-172 TO CONTINUOUS TABLE POSITIONS 1-144.
33. C
34. INDEX=INDKS
35. IF (INDEX.GT.100) INDEX=INDEX-28
36. C
37. C IF INDEX IS OUT OF RANGE, RETURN VALUE.
38. C
39. C IF (INDEX.GE.1.AND.INDEX.LE.144) GO TO 101
40. PRINT 102,INDKS
41. UNITS(1)=UNITLE
42. UNITS(2)=ERROR
43. RETURN
44. C
45. C CALCULATE LABEL FOR APPROPRIATE UNIT-CODE TABLE SUBROUTINE.
46. C
47. C IUI IF (TABLE(INDEX).EQ.0) RETURN
48. ITHPCD = TABLE(INDEX) / 100
49. INDEX = TABLE(INDEX) - ITHPCD * 100
50. IF (UNICDE .EQ. C) UNICDE = - ITHPCD
51. C
52. C BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
53. C
54. GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18),INDEX

```

55.	C	
56.	C	'LINEAR' TABLE.
57.	C	
58.	C	1 CALL LINEAR(VALUE,UNICODE,UNITS)
59.	C	RETURN
60.	C	
61.	C	'PLANAR' TABLE.
62.	C	
63.	C	2 CALL PLANAR(VALUE,UNICODE,UNITS)
64.	C	RETURN
65.	C	
66.	C	'VOLUME' TABLE.
67.	C	
68.	C	3 CALL VOLUME(VALUE,UNICODE,UNITS)
69.	C	RETURN
70.	C	
71.	C	'SHORT TIME' TABLE.
72.	C	
73.	C	4 CALL SHORTTIME(VALUE,UNICODE,UNITS)
74.	C	RETURN
75.	C	
76.	C	'LONG TIME' TABLE.
77.	C	
78.	C	5 CALL LONGTIME(VALUE,UNICODE,UNITS)
79.	C	RETURN
80.	C	
81.	C	'VELOCITY' TABLE.
82.	C	
83.	C	6 CALL SPEEDS(VALUE,UNICODE,UNITS)
84.	C	RETURN
85.	C	
86.	C	'TEMPERATURE' TABLE.
87.	C	
88.	C	7 CALL TEMPERATURE(VALUE,UNICODE,UNITS)
89.	C	RETURN
90.	C	
91.	C	'ANGLE' TABLE.
92.	C	
93.	C	8 CALL ANGLES(VALUE,UNICODE,UNITS)
94.	C	RETURN
95.	C	
96.	C	'PER CENT' TABLE.
97.	C	
98.	C	9 CALL PERCENT(VALUE,UNICODE,UNITS)
99.	C	RETURN
100.	C	
101.	C	'PRESSURE' TABLE.
102.	C	
103.	C	10 CALL PRESSURE(VALUE,UNICODE,UNITS)
104.	C	RETURN
105.	C	
106.	C	'ACOUSTIC PRESSURE' TABLE.
107.	C	
108.	C	11 CALL ACPRESSURE(VALUE,UNICODE,UNITS)
109.	C	RETURN
110.	C	
111.	C	'FREQUENCY' TABLE.



```

112. C
113. 12 CALL CYCLES(VALUE,UNICODE,UNITS)
114. RETURN
115. C
116. 'ENERGY' TABLE.
117. C
118. 13 CALL ENERGY(VALUE,UNICODE,UNITS)
119. RETURN
120. C
121. 'POWER' TABLE.
122. C
123. 14 CALL POWER(VALUE,UNICODE,UNITS)
124. RETURN
125. C
126. 'INTENSITY' TABLE.
127. C
128. 15 CALL INTENSITY(VALUE,UNICODE,UNITS)
129. RETURN
130. C
131. 'PRESSURE LEVEL' TABLE.
132. C
133. 16 CALL PRESSURE(VALUE,UNICODE,UNITS)
134. RETURN
135. C
136. 'DENSITY' TABLE.
137. C
138. 17 CALL DENSITY(VALUE,UNICODE,UNITS)
139. RETURN
140. C
141. 'DEEBEE' TABLE.
142. C
143. 18 CALL DEEBEE(VALUE,UNICODE,UNITS)
144. RETURN
145. 102 FORMAT(1, 'ERROR--VARIABLE NAME CODE',13,1 ' OUT OF RANGE.1')
146. END

```

WELT,L N.COPAVA,.COPAVA

SUBROUTINE COPAVA (CHDTYP,SEQ,CARDIN)

DATA IS READ AND STORED INTO THE APPROPRIATE SPOTS IN COMMON

CPV1

3/12/74 GFA

IMPLICIT INTEGER (A-Z)

COMMON /CPV /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,VAR(3,30),

NDATS,NMOMS(50),NDATA,VALMNG(2,30,50),DATA(30000)

DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50)

EQUIVALENCE (CON(1),ICON(1)),(PAR(1),IPAR(1,1)),

(VALMNG(1,1),IVLRNG(1,1,1))

REAL CON,PAR,DATA,VALMNG

DIMENSION CARDIN(14),FIELD(2),UNITS(4)

DATA BLNK1/' '

DATA BLNK2/' '

DATA NULL /07777777777777776/

DATA B1/'R '

DATA B2/'B '

DATA FIELD1/' '

DATA FIELD2/' '

INITIALIZE

IF (SEQ.NE. '20') GO TO 30

--PROCESS CONSTANTS--

DECODE NO. OF CONSTANTS

DECODE (20,CARDIN) MCON

20 FORMAT (T14,13)

CHECK FOR ERROR IN MCON

IF (MCON.GT. 0) GO TO 18

WRITE (6,17) (CARDIN(I),I=1,14)

17 FORMAT (' ERROR -- MCON =',I4,13A6,A2)

STOP

55.	C	CHECK FOR CONTINUATION
56.	C	
57.	C	18 IF INCON .GT. 4) GO TO 25
58.	C	
59.	C	
60.	C	DECODE (21,CARDIN) ((CON11,J),1=1,4),J=1,NCON1
61.	C	21 FORMAT (11,13,127,13,120,F7.0,130,213,143,13,136,F7.0,
62.	C	146,213,159,13,152,F7.0,162,213,175,13,168,F7.0,
63.	C	178,13)
64.	C	GO TO 24
65.	C	
66.	C	
67.	C	25 DECODE (21,CARDIN) ((CON11,J),1=1,4),J=1,4
68.	C	READ (9,21) ((CON11,J),1=1,4),J=5,NCON1
69.	C	24 CALL CDRD (CARDIN,SEQ,CRDTYP)
70.	C	
71.	C	
72.	C	CHECK FOR PARAMETERS
73.	C	
74.	C	30 IF (SEQ .NE. 30) GO TO 40
75.	C	
76.	C	--PROCESS PARAMETERS--
77.	C	
78.	C	INCREMENT NO. OF PARAMETERS COUNTER
79.	C	
80.	C	MPAR = MPAR + 1
81.	C	
82.	C	DECODE NAME,UNIT,PROCESS & NO. OF VALUES
83.	C	
84.	C	DECODE (32,CARDIN) (PAR(1,MPAR),1=1,4)
85.	C	32 FORMAT (114,413)
86.	C	
87.	C	
88.	C	NVAL = IPAR(1,MPAR)
89.	C	IF (NVAL .GT. 6) GO TO 35
90.	C	
91.	C	NO. DO THIS CARD
92.	C	
93.	C	J= NVAL + 4
94.	C	DECODE (33,CARDIN) (PAR(1,MPAR),1=5,J)
95.	C	33 FORMAT (127,6F9.0)
96.	C	GO TO 37
97.	C	
98.	C	
99.	C	35 DECODE (33,CARDIN) (PAR(1,MPAR),1=5,10)
100.	C	READ (9,33) (PAR(1,MPAR),1=1,J)
101.	C	
102.	C	READ NEXT CARD
103.	C	
104.	C	37 CALL CDRD (CARDIN,SEQ,CRDTYP)
105.	C	
106.	C	CONTINUE TO PROCESS ALL PARAMETERS
107.	C	
108.	C	GO TO 30
109.	C	
110.	C	
111.	C	

```

112.      40 IF (SEQ .EQ. '4C') GO TO 45
113.      C
114.      NO VARIABLES, CHECK FOR E ON A TYPE
115.      C
116.      IF (FLDIC,6,CARDIN(1)) .EQ. 'E') RETURN
117.      C
118.      A TYPE, WRITE ERROR
119.      C
120.      WRITE (6,41) (CARDIN(1),1-1,14)
121.      41 FORMAT (//,1, ' ERROR -- NO AN40 CARD ENCOUNTERED WHEN EXPECTED.',/,
122.      *
123.      * 1X,13A6,A2)
124.      STOP
125.      C
126.      --PROCESS VARIABLES--
127.      C
128.      DECODE # OF VARIABLES
129.      C
130.      45 DECODE (20,CARDIN) NVAR
131.      C
132.      IF (NVAR .GT. 7) GO TO 47
133.      C
134.      NO CONT., DO THIS CARD
135.      C
136.      DECODE (46,CARDIN) ((1VAR(1),1-1,1,3),J01,NVAR)
137.      46 FORMAT (110,7(1,1))
138.      GO TO 50
139.      C
140.      C
141.      47 DECODE (56,CARDIN) ((1VAR(1),1-1,1,3),J01,1,2)
142.      READ (9,48) ((1VAR(1),1-1,1,3),J08,NVAR)
143.      48 FORMAT (127,4(1,1))
144.      C
145.      READ NEXT CARD
146.      C
147.      50 CALL CRORED (CARDIN,569,CROTYPE)
148.      C
149.      CHECK FOR DATA SET CARD
150.      C
151.      IF (SEQ .EQ. '50') GO TO 55
152.      C
153.      WRITE ERROR
154.      C
155.      WRITE (6,52) (CARDIN(1),1-1,14)
156.      52 FORMAT (//,1, ' ERROR -- AN OR EN 50 CARD NOT ENCOUNTERED WHEN EXPECTE
157.      * 0.',/11X,13A6,A2)
158.      C
159.      STOP
160.      C
161.      --PROCESS DATA SETS CARD(S)--
162.      C
163.      DECODE NUMBER OF DATA SETS
164.      C
165.      55 DECODE (20,CARDIN) NDATS
166.      C
167.      NDATSC =1
168.

```



```

169. IF (NPAR.EQ. 0) GO TO 60
170. DO 57 I=1,NPAR
171.   NDATSC = NDATSC + 1*PART(I,1)
172.   57 CONTINUE
173.   C
174.   C CHECK BOTH THE SAME
175.   C
176.   60 IF (INDATS.EQ. NDATSC) GO TO 65
177.   C
178.   C WRITE ERROR
179.   C
180.   WRITE (6,61) NDATS,NDATSC,(CARDIN(I),I=1,14)
181.   61 FORMAT (' ERROR -- THE NUMBER OF DATA SETS (',13,') COUNTED DID NO
182.   *T AGREE WITH',/,11, 'THE NUMBER OF DATA SETS (',13,') COMPUTED.',/,
183.   *1X,13A6,A2)
184.   C
185.   C STOP
186.   C
187.   C
188.   65 IF (INDATS.GT. 16) GO TO 70
189.   C
190.   C DECODE THIS CARD
191.   C
192.   C [CODE (46,CARDIN) (NR0MS(I),I=1,NDATS)
193.   66 FORMAT (17,1614)
194.   GO TO 75
195.   C
196.   C
197.   70 DECODE (66,CARDIN) (NR0MS(I),I=1,16)
198.   READ (9,66) (NR0MS(I),I=17,NDATS)
199.   C
200.   C
201.   75 DO 77 I=1,NDATS
202.   IF (NR0MS(I).GT. 0) GO TO 77
203.   WRITE (6,76) I,NR0MS(I),(CARDIN(J),J=1,14)
204.   76 FORMAT (' ERROR -- NR0MS(',12,') .LT. 1,110,/,1X,13A6,A2)
205.   STOP
206.   77 CONTINUE
207.   C
208.   C
209.   C 80 NDATA = 0
210.   C
211.   C PROCESS ALL DATA SETS
212.   C
213.   C
214.   C DO 100 L=1,NDATS
215.   C
216.   C COMPUTE NC, REMAINING & INIT COUNTER
217.   C
218.   C N0REM = NR0MS(L) + NVAR
219.   C NCONT = 0
220.   C
221.   C
222.   C 85 CALL MOVINT(CARDIN,21,10)
223.   C
224.   C READ A CARD
225.   C

```

```

226. CALL CDRD (CARDIN,SEQ,CRDTYP)
227. C
228. C
229. HONC = 0
230. C
231. C
232. MDL = 6
233. IF (NOREM .LT. 6) MDL = NOREM
234. C
235. C
236. C
237. C
238. C
239. C
240. HONC = HONC + 1
241. HCONT = HCONT + 1
242. IF (.NOT. (FIELD(1) .EQ. BLNK1 .AND. FIELD(2) .EQ. BLNK2))
243. * GO TO 95
244. C
245. C
246. C
247. C
248. WRITE (6,92) HONC, (CARDIN(1)), (1,1,14)
249. 92 FORMAT (//, ' ERROR-- BLANK DATA FIELD (',1,1,') ENCOUNTERED.',/,/,
250. * 1X,12A6,A2)
251. STOP
252. C
253. C
254. C
255. C
256. C
257. C
258. C
259. C
260. C
261. C
262. C
263. C
264. C
265. C
266. C
267. C
268. C
269. C
270. C
271. C
272. C
273. C
274. C
275. C
276. C
277. C
278. C
279. C
280. C
281. C
282. C

```

(CALL CDRD (CARDIN,SEQ,CRDTYP)  
 HONC = 0  
 MDL = 6  
 IF (NOREM .LT. 6) MDL = NOREM  
 MOVE NEXT SET OF CHARACTERS  
 90 CALL MOVCHN(FIELD(1))  
 HONC = HONC + 1  
 HCONT = HCONT + 1  
 IF (.NOT. (FIELD(1) .EQ. BLNK1 .AND. FIELD(2) .EQ. BLNK2))  
 \* GO TO 95  
 BLANK DATA FIELD  
 WRITE (6,92) HONC, (CARDIN(1)), (1,1,14)  
 92 FORMAT (//, ' ERROR-- BLANK DATA FIELD (',1,1,') ENCOUNTERED.',/,/,  
 \* 1X,12A6,A2)  
 STOP  
 CHECK FOR 'NULL' DATA FIELD  
 95 NDATA = NDATA + 1  
 DATA(NDATA) = NULL  
 IF (FIELD(1) .EQ. 01 .AND. FIELD(2) .EQ. 02) GO TO 98  
 DECODE NUMERIC VALUE  
 DECODE (97, FIELD) DATA(NDATA)  
 97 FORMAT (F10.0)  
 LL = MOD(HCONT, NVAR)  
 IF (LL .EQ. 0) LL = NVAR  
 CALL CONVRT (DATA(NDATA), IVAR(1,LL), IVAR(2,LL), UNITS)  
 IF (UNITS(1) .EQ. 'UNITS ') WRITE (6,96) (CARDIN(1), 10, 14)  
 96 FORMAT (1X,12A6,A2)  
 CHECK FOR END OF CARD  
 98 IF (HONC .NE. MDL) GO TO 90  
 NOREM = NOREM - MDL  
 CHECK FOR END OF DATA SET

```

283.      IF (NOREM .GT. 0) GO TO B5
284.      C
285.      C
286.      C 100 CONTINUE
287.      C
288.      C
289.      C
290.      IF (NCON .EQ. 0) GO TO 115
      DO 110 I=1,NCON
291.      CALL CONVRT (CON(3,1),CON(1,1),CON(2,1),UNITS)
292.      IF (UNITS(1).EQ.'UNITS') WRITE(6,107) (CON(J,1),J=1,3)
293.      107 FORMAT(' CONSTANTS ',215,F21.7)
294.      110 CONTINUE
295.      C
296.      C
297.      115 IF (NPAR .EQ. 3) GO TO 150
298.      DO 130 I=1,NPAR
299.      K = IPAR(1,1)
300.      C
301.      DO 120 J=1,K
302.      CALL CONVRT (PAR(1+J,1),IPAR(2,1),IPAR(3,1),UNITS)
303.      IF (UNITS(1).EQ.'UNITS') WRITE(6,117) PAR(4+J,1),
304.      *IPAR(2,1),IPAR(3,1)
305.      117 FORMAT(' PARAM. ',F21.7,215)
306.      120 CONTINUE
307.      C
308.      C 130 CONTINUE
309.      C
310.      150 RETURN
311.      C
312.      C ALL DONE
313.      C
314.      C END

```

QFOR,SA N,CNRED, CNRED

```
1. SUBROUTINE CNRED (CARDIN,SEQ,CROTYP)
2. C
3. C CNRED (CARD READ) READS 80 COLUMNS OF THE NEXT CARD AND
4. C PUTS IT IN CARDIN. IT ALSO STORES COL 162 IN CROTYP & 263
5. C IN SEQ.
6. C
7. C DIMENSION CARDIN(1)
8. C
9. C READ (9,10) (CARDIN(1),1,1,14)
10. C IO FORMAT (13A6,A2)
11. C
12. C FLD(1,12,SEQ) = FLD(1,12,CARDIN(1))
13. C FLD(1,12,CROTYP) = FLD(1,12,CARDIN(1))
14. C
15. C RETURN
16. C END
```



9F04.2- N.CRUINT,CRUINT

```
1. SUBROUTINE CRUINT
2. C
3. C CRUINT (CRUISE INITIALIZED) INITIALIZES THE COMMON
4. C /CRUISE/ EVERY TIME A NEW CRUISE RECORD IS ENCOUNTERED.
5. C
6. C
7. C IMPLICIT INTEGER (A-Z)
8. C
9. C
10. C COMMON /CRUISE/CRUNO,PNATC,CALAT,CALON,CAIDAT,CAFLAT,CAFLON,
11. C CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
12. C CASTATZT,CESTATZT,PACS,PENS,NCARDS,NOTES114,1321,
13. C CAFLG
14. C
15. C
16. C
17. C PNATC = 0
18. C CAFLG = 0
19. C CALON = -179600
20. C CAIDAT = 991232
21. C CAFLAT = 90000
22. C CAFLON = 179600
23. C CAFDAT = 0
24. C
25. C CEILAT = -90000
26. C CEILON = -179600
27. C CEIDAT = 991232
28. C CEFLAT = 90000
29. C CEFLON = 179600
30. C CEFDAT = 0
31. C
32. C RETURN
33. C
34. C
```

GET,L N,CRUSUM,CRUSUM

1\* SUBROUTINE CRUSUM (BFR1,BFR2)

2\* C

3\* C

4\* C

5\* IMPLICIT INTEGER (A-Z)

6\* C

7\* C

8\* COMMON /CRUISE/CRUNO,PNATC,CALAT,CALON,CAIDAT,CAFLAT,CAFLON,  
9\* CAPDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,  
10\* CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),  
11\* CAFLG

12\* C

13\* C

14\* C

15\* COMMON /ASTATN/SANO,SANRUM,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,  
16\* SAFDAT,SASTAT(2),PNXTAS,PENVS,SAIFLG

17\* C

18\* C

19\* C

20\* C

21\* COMMON /SECTO / ID(17),IDATE,MXISEC,LSTADR,LEXP,MOCRU,MOSTA,  
22\* LSTADOC,LSTADA,LSTADE

23\* C

24\* C

25\* C

26\* C

27\* FLD(18,18,SECADR) = FLD(18,18,LSTADOC)

28\* FLD(18,18,MSECI) = FLD(18,18,LSTADOC)

29\* C

30\* CALL DSKRED (SECADR,MSECI,BFR1)

31\* C

32\* UNPACK IT

33\* C

34\* CALL UNPKCR (BFR1)

35\* C

36\* FLD(18,18,SECADR) = FLD( 0,18,LSTADA)

37\* FLD(18,18,MSECI) = FLD(18,18,LSTADA)

38\* C

39\* CALL DSKRED (SECADR,MSECI,BFR2)

40\* C

41\* UNPACK IT

42\* C

43\* CALL ASUNPK (BFR2,TYPE)

44\* C

45\* IF (SAFLAT .GT. CALLAT) CALAT = SAFLAT

46\* IF (SAFLAT .GT. CALAT) CALAT = SAFLAT

47\* IF (SAFLAT .LT. CAFLAT) CAFLAT = SAFLAT

48\* IF (SAFLAT .LT. CAFLAT) CAFLAT = SAFLAT

49\* C

50\* IF (CAFLG .NE. 0) GO TO 50

51\* CALON = EAST(SAILON,SAFLON)

52\* CAFLON = WEST(SAILON,SAFLON)

53\* GO TO 60

54\* C

```

55. 50 CAILON = EAST(SAILON,CAILON)
56. CAFLON = WEST(SAFLON,CAFLON)
57. C
58. 60 IF (SAIDAT .LT. CAIDAT) CAIDAT = SAIDAT
59. IF (SAFDAT .GT. CAFDAT) CAFDAT = SAFDAT
60. C
61. CASTAT(1) = ORICASTAT(1),SASTAT(1)
62. CASTAT(2) = ORICASTAT(2),SASTAT(2)
63. C
64. FLD(10,10,SECAD1) = FLD( 0,10,1,STADE)
65. FLD(10,10,MSEC1) = FLD(10,10,1,STADE)
66. C
67. CALL DSKRED (SECAD1,MSEC1,BFR2)
68. C
69. C UNPACK IT
70. C
71. CALL ASUMPK (BFR2,TYPE)
72. C
73. IF (SAILAT .GT. CEILAT) CEILAT = SAILAT
74. IF (SAFLAT .GT. CEFLAT) CEFLAT = SAFLAT
75. IF (SAILAT .LT. CEFLAT) CEFLAT = SAILAT
76. IF (SAFLAT .LT. CEFLAT) CEFLAT = SAFLAT
77. C
78. IF (CAFLG .NE. Q) GO TO 100
79. CAFLG = I
80. CEILON = EAST(SAILON,SAFLON)
81. CEFLON = WEST(SAILON,SAFLON)
82. GO TO 110
83. 100 CEILON = EAST(SAILON,CEILON)
84. CEFLON = WEST(SAFLON,CEFLON)
85. C
86. 110 IF (SAIDAT .LT. CEIDAT) CEIDAT = SAIDAT
87. IF (SAFDAT .GT. CEFDAT) CEFDAT = SAFDAT
88. C
89. CESTAT(1) = ORICESTAT(1),SASTAT(1)
90. CESTAT(2) = ORICESTAT(2),SASTAT(2)
91. C
92. C REPACK CRUISE
93. C
94. CALL PCKCRU (BFR1,M)
95. C
96. C
97. CALL DSKRIT (SECADR,MSEC,BFR1)
98. C
99. C DONE
100. C
101. RETURN
102. END

```

N. CYCLES, CYCLES

```

1. SUBROUTINE CYCLES(VALUE,UNICODE,UNITS)
2.
3. C TABLE OF CYCLES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER (ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,-1,3)/77,70,79/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1,-2,3),-1,3)/,HERTZ , ,
18. C ,
19. C , 'KILOHERTZ',RTZ , , 'C.P.M.', ,
20. C
21. C ENTER INTERNAL UNITS.
22. C
23. C UNITS(3)= 'HERTZ' ,
24. C UNITS(4)= ,
25. C
26. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
27. C
28. C IF(UNICODE.EQ.0) UNICDE=-77
29. C
30. C SET CONVERSION FACTORS.
31. C
32. C DATA (FACTOR(1),1,1,3)/1.000,1.003,4.001/
33. C
34. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
35. C CODE INTO 'UNITS'.
36. C
37. C ICODE=IABS(UNICODE)
38. C DO 2 I=1,3
39. C IF(ICODE.NE.ENTRY(1,I)) GO TO 2
40. C IF(UNICODE.EQ.0) VALUE=VALUE*FACTOR(I)
41. C IF(UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
42. C DO 1 J=1,2
43. C UNITS(J)=ENTRY(J+1,1)
44. C 1 CONTINUE
45. C
46. C AT THIS POINT, CONVERSION IS COMPLETE.
47. C
48. C RETURN
49. C
50. C ELSE, CHECK REST OF TABLE.
51. C
52. C 2 CONTINUE
53. C
54. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
55. C

```



55. UNITS(1)=UNITS \*  
56. UNITS(2)=ERROR \*  
57. PRINT 101,ICODE  
58. ICI FORMAT(' ERROR--UNIT CODE',13,' NOT IN FREQUENCY TABLE.')

59. RETURN  
60. END

QFUR,5m N,DBLOSS,DOBLOSS

```

1. SUBROUTINE DBLOSS(VALUE,UNICODE,UNITS)
2. C
3. C TABLE FOR DB LOSS AND ATTENUATION CONVERSION FACTORS.
4. C
5. C DIMENSION FACTOR(8)
6. C INTEGER ENTRY(3,8),UNITS(4),UNICODE
7. C
8. C PUT UNIT CODES INTO ENTRY.
9. C
10. C DATA (ENTRY(1,1),1=1,8)/07,01,02,90,92,93,94,95/
11. C
12. C ENTER CONVERSION FACTORS.
13. C
14. C DATA (FACTOR(1,1)=1,8)/.0,-59.2,0.0,-60.0,0.0,.0,59.2,60.0/
15. C
16. C PUT IN ALPHA UNITS.
17. C
18. C DATA ((ENTRY(1,J),1=2,3),J=1,8)/08//1,0,YARD,0,
19. C 'DB//1,0,KYARD,0,08//1,0,METER,0,
20. C 'DB//1,0,KMETER,0,08/MET,0,PER,0,
21. C 'DB/YAN,0,D,0,08/KYAN,0,RO,0,
22. C 'DB/KME,0,TER,0,
23. C
24. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
25. C
26. C UNITS(1)='UNITS'
27. C UNITS(2)='ERROR'
28. C
29. C RETRIEVAL IN STANDARD DB LOSS UNITS.
30. C
31. C IF(UNICODE.NE.0) GO TO 1
32. C UNICDE=-82
33. C
34. C ENTER INTERNAL UNITS.
35. C
36. C UNITS(3)='DB//1'
37. C UNITS(4)='METER'
38. C GO TO 2
39. C
40. C RETRIEVAL IN STANDARD ATTENUATION UNITS.
41. C
42. C 1 IF(UNICODE.NE.-1) GO TO 2
43. C UNICDE=-92
44. C
45. C ENTER INTERNAL UNITS.
46. C
47. C UNITS(3)='DB/MET'
48. C UNITS(4)='ER'
49. C
50. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
51. C CODE INTO 'UNITS'.
52. C
53. C 2 ICODE=IABS(UNICODE)
54. C 00 3 1=1.8

```

```

55. IF (ICODE=NE.ENTRY(1,1)) 6J TO 3
56. IF (UNICDE.GT.0) VALUE=VALUE*FACTOR(1)
57. IF (UNICDE.LT.0) VALUE=VALUE*FACTOR(1)
58. UNITS(1)=ENTRY(2,1)
59. UNITS(2)=ENTRY(3,1)
60. C
61. C AT THIS POINT, CONVERSION IS COMPLETE.
62. C
63. RETURN
64. 3 CONTINUE
65. C
66. C ELSE, UNIT CODE IS NOT CONVERTIBLE.
67. C
68. PRINT IDL,ICODE
69. PRINT 112
70. 112 FORMAT(' ERROR--UNIT CODE NOT CONVERTIBLE!')
71. UNITS(1)=UNITS *
72. RETURN
73. 101 FORMAT(' ERROR--UNIT CODE',13,' NOT IN 00 LOSS TABLE.')
74. END

```

33333001 3983 10 N

SUBROUTINE DEEBE (VALUE, UNICDE, UNITS)

TABLE OF CU UNITS.

INTEGER ENTRY(3.27),UNIT5(4),UNICDE

PUT CODES INTO ENTRY.

DATA (ENTRY(1,1),I=1,271/39,40,41,58,66,67,68,69,72,73,83,86,96,

0011-0110/95/0004-0000\$05.00/0

ENTEN ALPHA UNITS:

MV018B7EMJ0...ZM/S/Z0...MMII//  
Z010D1/BQ../7/7'1oc'(C'1)YMINI)VIV

PRESS, R. RAY, 1 DB//1W, 1 ALL  
 DB//1W, 1 M002, 1 DB//1W, 1 CH002, 1

0.7M=2M30.0/311//0.0, M3/33.0.5/5031.0  
0.7M3/3M.1.0.31//00.1.7M3/35.1.0M31//.

'DB ,  
,//ENG//,CHZBTT,  
'DB ,  
,IDB//I ,IKYABD ,

```

'DB//I',METER 'DB//I',METER,
'DB/MET',METER 'DB/MET',METER,
'DB/YAM',METER 'DB/YAM',METER,

```

DB/KVA, NO  
DB/KVA, NO  
DB/KVA, NO  
DB/KVA, NO

068//10'. PABIM V. 068//10'. PABIVB. 1//108A. 182501V. 1//108A. 12 501M. 1

IS UNICEF A C.O. THEN SET UNITS TO 800 AS MEAS. B.

CHAPTER 10

• SY3W • (2) SL1MN

SET INDICATOR FOR CASE UNPACK IS BYPASSED.

0-1511

IF RETRIEVING, UNPACK VALUE AND UNIT CODE.

IF (UNICODE.1) GO TO 1

```
IF(UNICODE.EQ.0.AND.ICODE.EQ.-99999999) RETURN
```

UNICDE# 03031UN

[illegible][illegible][illegible]

100

[illegible]



```

55. C TEST TO SEE IF CODE IS IN TABLE.
56. C
57. 1 ICODE=TAGS(UNITCODE)
58. DO 2 I=1,27
59. IF (ICODE.EQ.ENTRY(I,1)) GO TO 3
60. 2 CONTINUE
61. C
62. C IF CODE NOT FOUND, WRITE ERROR.
63. C
64. PRINT 101,ICODE
65. PRINT 102,UNITCODE,VALUE,ITST
66. 102 FORMAT(' UNITCODE=',I5,' VALUE(OCTAL)=' ,O12,' ITST=',I5)
67. UNITS(1) = 'UNITS'
68. UNITS(2) = 'ERROR'
69. RETURN
70. C
71. C PACK CODE AND VALUE, IF NECESSARY.
72. C
73. 3 IF (ITST.EQ.1) GO TO 4
74. FLD(29,7,VALUE)=FLD(29,7,UNITCODE)
75. C
76. C SET UNITS, RETURN.
77. C
78. 4 UNITS(1)=ENTRY(2,1)
79. UNITS(2)=ENTRY(3,1)
80. UNITS(3)=ENTRY(2,1)
81. UNITS(4)=ENTRY(3,1)
82. RETURN
83. 101 FORMAT(' ERROR--UNIT CODE=',I3,' NOT IN DEEBEE TABLE.')
84. END

```

QFOR,SW N,DENSTY,,DENSTY

```

1. SUBROUTINE DENSITY(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF DENSITY MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,2),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(1)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA IENTRY(1,1),I(1,2),I(1,3)/45,46,47/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA IENTRY(1,2),I(1,3),I(1,4)/'GMS/CM','.003',
18. C 'KG/MET','.003','.LB/FT','.003' /
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(1)='GMS/CM'
23. C UNITS(2)='.003'
24. C
25. C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF(UNICDE.EQ.0) UNICDE=-40
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA IFACOR(1),I(1,2),I(1,3)/1.000,1.000,7.76922304/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C UNITS(1)='UNITS'
35. C UNITS(2)='ERROR'
36. C
37. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
38. C CODE INTO 'UNITS'.
39. C
40. C ICODE=IABS(UNICDE)
41. C DO 2 I=1,3
42. C IF(ICODE.NE.ENTRY(1,I)) GO TO 2
43. C IF(UNICDE.EQ.0) VALUE=VALUE*FACTOR(1)
44. C IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(1)
45. C UNITS(1)=ENTRY(2,I)
46. C UNITS(2)=ENTRY(3,I)
47. C
48. C AT THIS POINT, CONVERSION IS COMPLETE.
49. C
50. C RETURN
51. C
52. C ELSE, CHECK REST OF TABLE.
53. C
54. C 2 CONTINUE

```

```

55. C
56. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
57. C
58. PRINT I3I, ICODE
59. I3I FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN DENSITY TABLE. ')
60. RETURN
61. END

```

QELT,L H,OSKRED,,OSKRED

1. SUBROUTINE OSKRED(IN,IS,ADR)

2. C

3. C

4. CALL SETADR(10,IN)

5. IW = IS \* 28

6. CALL NTRAN(10,2,IW,ADR,L)

7. CALL NTRAN(10,22)

8. C

9. IF (L .GT. 0) RETURN

10. C

11. WRITE (6,10) IN,IS,L

12. 10 FORMAT(' ',ERROR IN OSKRED',3T10)

13. RETURN 0

14. END



DELTY,L N,DSKRIT,,DSKRIT

1. SUBROUTINE DSKRIT(IN,IS,ADR)

2. C

3. C

4. CALL SETADR(IJ,INH)

5. IN = IS \* 24

6. CALL NTRAN(IG,1,IN,ADR,L)

7. CALL NTRAN(IG,22)

8. C

9. IF (L .GT. 0) RETURN

10. C

11. WRITE (4,10) IN,IS,L

12. 10 FORMAT('ERROR IN DSKRIT',J113)

13. RETURN 0

14. END

WLT,L N,EAST,EAST

```
1: C
2: C
3: C
4: C
5: C
6: INTEGER A,B
7: IF (ABS(A-B) .GT. 180000) GO TO 10
8: EAST = MAX(A,B)
9: RETURN
10: EAST = B
11: IF (A .LE. 0) EAST = A
12: RETURN
13: END
```

DEFORM,SA N,ENERGY,ENERGY

```

1. SUBROUTINE ENERGY(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ENERGY MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(1,5),UNITS(1),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(5)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,1,5)/(91.62,63.09,897
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA (ENTRY(1,1),1,1,2,3),(1,1,5),(1,1,5) /
18. C
19. C 'JOULES', 'FT-LB', 'S',
20. C
21. C 'KGM ME', 'TERS', 'AMPIU'S',
22. C
23. C ENTER INTERNAL UNITS.
24. C
25. C UNITS(1)=KGM ME.
26. C
27. C UNITS(4)=TERS
28. C
29. C IF UNICDE=G, THEN SET CODE TO STANDARD UNITS.
30. C
31. C IF (UNICDE.EQ.U) UNICDE=-88
32. C
33. C SET CONVERSION FACTORS.
34. C
35. C DATA (FACTOR(1),1,1,5)/(1.01970D-8,1019700,138255500,100,107.500/
36. C
37. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
38. C
39. C UNITS(1)=UNITS
40. C
41. C UNITS(2)=ERROR
42. C
43. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
44. C
45. C CODE INTO 'UNITS'.
46. C
47. C ICODE=TABLES(UNICODE)
48. C
49. C DO 2 1=1,5
50. C
51. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
52. C
53. C IF (UNICDE.GE.0) VALUE=VALUE*FACTOR(1)
54. C
55. C IF (UNICDE.LT.C) VALUE=VALUE/FACTOR(1)
56. C
57. C UNITS(1)=ENTRY(2,1)
58. C
59. C UNITS(2)=ENTRY(3,1)
60. C
61. C AT THIS POINT, CONVERSION IS COMPLETE.
62. C
63. C RETURN
64. C
65. C ELSE, CHECK REST OF TABLE.

```

55. C  
56. 2 CONTINUE  
57. C  
58. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.  
59. C  
60. PRINT I01,ICODE  
61. I01 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ENERGY TABLE.')

RETURN

END



QELT,L N,ENSTAT,,ENSTAT

1. SUBROUTINE ENSTAT (STATUS)

2. C  
3. C  
4. C  
5. C  
6. C  
7. C  
8. C  
9. C  
10. C  
11. C  
12. C  
13. C  
14. C  
15. C  
16. C  
17. C  
18. C  
19. C  
20. C  
21. C  
22. C  
23. C  
24. C  
25. C  
26. C  
27. C  
28. C  
29. C  
30. C  
31. C  
32. C  
33. C  
34. C  
35. C  
36. C  
37. C  
38. C  
39. C  
40. C  
41. C  
42. C  
43. C  
44. C  
45. C  
46. C  
47. C  
48. C  
49. C  
50. C  
51. C  
52. C  
53. C  
54. C

IMPLICIT INTEGER (A-Z)

COMMON /CPV /NCON,CONT4,301,NPAR,PAR(30,30),NVAR,IVAR(3,30),

NDATS,NROWS(50),NUATA,VALRNG(2,30,50),DATA(300001

DIMENSION ICON(4,30),IPAR(30,30),IVLNG(2,30,50)

EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),

(VALRNG(1,1,1),IVLNG(1,1,1))

REAL CON,PAR,DATA,VALRNG

DIMENSION STATUS(2)

CHECK FOR CONSTANTS

IF (NCON .EQ. 0) GO TO 50

DO 40 I=1,NCON

IF (ICON(I,1) .LT. 10) GO TO 7

ICOD = ABS(ICON(I,1) - 101)

ISUB = 1

IF (ICOD .LE. 35) GO TO 10

ISUB = 2

ICOD = ABS(ICOD - 36)

IF (ICOD .LE. 35) GO TO 10

7 WRITE (6,4) ICON(I,1)

4 FORMAT (1 'ERROR -- ENSTAT - ILLEGAL CODE',110)

RETURN 0

10 FLD(ICOD,1,STATUS(ISUB)) = 1

40 CONTINUE

CHECK PARAMETERS

50 IF (NPAR .EQ. 0) GO TO 70

DO 65 I=1,NPAR

IF (IPAR(2,I) .LT. 10) GO TO 57

ICOD = ABS(IPAR(2,I) - 101)

ISUB = 1

IF (ICOD .LE. 35) GO TO 60

ISUB = 2

ICOD = ABS(ICOD - 36)

IF (ICOD .LT. 35) GO TO 60

57 WRITE (6,4) IPAR(2,I)

RETURN 0

```

55. C
56. 40 FLD(ICOD,1,STATUS(ISUB)) = 1
57. 65 CONTINUE
58. C
59. C CHECK VARIABLES
60. C
61. 70 IF INVAR .EQ. 0) GO TO 100
62. C
63. CC 90 I=1,NVAR
64. C
65. IF (IVAR(I,1) .LT. 101) GO TO 77
66. ICODE = ABS(IVAR(I,1) - 101)
67. C
68. ISUB = 1
69. IF (ICOD .LE. 35) GO TO 60
70. ISUB = 2
71. ICODE = ABS(ICOD - 36)
72. C
73. IF (ICOD .LT. 35) GO TO 80
74. 77 WRITE (6,4) IVAR(I,1)
75. RETURN 0
76. C
77. 80 FLD(ICOD,1,STATUS(ISUB)) = 1
78. 90 CONTINUE
79. C
80. 100 RETURN
81. END

```

```

QELT,L      N,(XPINT,,EXPINT
1.          SUBROUTINE EXPINT
2.          C
3.          IMPLICIT INTEGER (A-Z)
4.          C
5.          COMMON /EXPINT/EXNO,PNATE,PCRU,EAILAT,EAILON,EALDAT,EAFLAT,EAFLON,
6.          EAFDAT,EEFLAT,EEFLON,EEFDAT,
7.          EASSTAT(2),EESTAT(2),EAFLG
8.          C
9.          C
10.         C
11.         C
12.         C
13.         C
14.         C
15.         C
16.         C
17.         C
18.         C
19.         C
20.         C
21.         C
22.         C
23.         C
24.         C
25.         C
26.         C
27.         C
28.         C

```

PNATE = 0  
 PCRU = 0  
 EAILAT = -90000  
 EAILON = 0  
 EAFLG = 0  
 EALDAT = 991232  
 EAFLAT = 90000  
 EAFLON = 179600  
 EAFDAT = 0  
 EEILAT = -90000  
 EEILON = -179600  
 EELDAT = 991232  
 EEFLAT = 90000  
 EEFLON = 179600  
 EEFDAT = 0  
 RETURN  
 END

DEL T, L N, EXPSUM, .EXPSUM

1. SUBROUTINE EXPSUM (BFR1, BFR2)

2. C

3. C

4. C

5. IMPLICIT INTEGER (A-Z)

6. C

7. C

8. C

9. COMMON /EXPMT/EXNO, PNATE, PCRU, EAILAT, EAILON, EALDAT, EAFLAT, EAFLON, EAFDAT, EEILAT, EEILON, EEIDAT, EEFLAT, EEFLON, EEFDAT, EASTAT(2), EESTAT(2), EAFLG

10. C

11. C

12. C

13. C

14. COMMON /CRUISE/CRUNO, PNATC, CAILAT, CAILON, CAIDAT, CAFLAT, CAFLON, CAFDAT, CEILAT, CEILON, CEIDAT, CEFLAT, CEFLON, CEPDAT, CASTAT(2), CESTAT(2), PACS, PEMS, NCARDS, NOTES(14,132), CAFLG

15. C

16. C

17. C

18. C

19. C

20. COMMON /SECTO /IDIT, IDATE, NATSEC, LSTADR, LEXP, NOCRU, MOSTA, LSTADC, LSTADA, LSTADE

21. C

22. C

23. C

24. C

25. C

26. C

27. C

28. C

29. C

30. C

31. C

32. C

33. C

34. C

35. C

36. C

37. C

38. C

39. C

40. C

41. C

42. C

43. C

44. C

45. C

46. C

47. C

48. C

49. C

50. C

51. C

52. C

53. C

54. C



```

55.      IF (CAPDAT .GT. EAFDAT) EAFDAT = CAPDAT
56.      C
57.      EASTAT(1) = ORIEASTAT(1),CASTAT(1))
58.      EASTAT(2) = ORIEASTAT(2),CASTAT(2))
59.      C
60.      IF (CEILAT .GT. EEILAT) EEILAT = CEILAT
61.      IF (CEFLAT .GT. EEFLAT) EEFLAT = CEFLAT
62.      IF (CEILAT .LT. EEFLAT) EEFLAT = CEILAT
63.      IF (CEFLAT .LT. EEFLAT) EEFLAT = CEFLAT
64.      C
65.      IF (EAFLG .NE. 0) GO TO 100
66.      EAFLG = 1
67.      EAILON = EAST (CEILON,CEFLON)
68.      EAFLOM = WEST (CEILON,CEFLON)
69.      GO TO 110
70.      100 EAILON = EAST(CEILON,EEILON)
71.      EEFLOM = WEST(CEFLON,EEFLON)
72.      C
73.      110 IF (CEIDAT .LT. EEIDAT) CEIDAT = CEIDAT
74.      IF (SEEDAT .GT. EEFDAT) EEFDAT = SEEDAT
75.      C
76.      EESTAT(1) = ORIEESTAT(1),CESTAT(1))
77.      EESTAT(2) = ORIEESTAT(2),CESTAT(2))
78.      C
79.      C      NEPACK EXPERIMENT
80.      C
81.      CALL PCKEXP (BFRI,N)
82.      C
83.      C      WHITE EXPERIMENT RECORD BACK
84.      C
85.      CALL DSKRIT (SECADR,MSEC,BFRI)
86.      C
87.      C      DONE 1
88.      C
89.      RETURN
90.      END

```

WELT,L N,INITU,INITO

```

1: C
2: C
3:
4: DIMENSION D(28)
5: INTEGER D
6: D(1) = 1 NAVI
7: D(2) = 1 SEA UN
8: D(3) = 1 DENBAT
9: D(4) = 1 ER ACO
10: D(5) = 1 USTIC
11: D(6) = 1 DATA B
12: D(7) = 1 ANK
13: D(8) = 32674
14: D(9) = 1
15: D(10) = 0
16: D(11) = 0
17: D(12) = 0
18: D(13) = 0
19: D(14) = 0
20: D(15) = 0
21: D(16) = 0
22: C
23: CALL NTRAN(10,22)
24: CALL SETADR(10,0)
25: CALL NTRAN(10,1,20,0,0,LSTAT)
26: C
27: IF LSTAT .GT. 01 STOP
28: C
29: WRITE (4,10) LSTAT
30: 10 FORMAT (1 ENROR -- ,15)
31: STOP
32: END

```

OFOR,SA N,INTENS,,INTENS

```

1: SUBROUTINE INTENS(VALUE,UNICODE,UNITS)
2: C
3: C TABLE OF INTENSITY LEVEL CONVERSION FACTORS.
4: C
5: C INTEGER ENTRY(1,2,3),UNITS(4),UNICODE
6: C
7: C FACTORS ARE DOUBLE PRECISION.
8: C
9: C DOUBLE PRECISION FACTOR(3)
10: C
11: C PUT UNIT CODES INTO ENTRY.
12: C
13: C DATA (ENTRY(1,1),1=1,3)/74,75,76/
14: C
15: C PUT IN ALPHA UNITS.
16: C
17: C DATA (ENTRY(1,1),1=2,3)/1,3)/:WATTS/CM,1,002,1,
:WATTS/CM,1,002/HZ/
18: C
19: C ENTER INTERNAL UNITS.
20: C
21: C UNITS(3)=WATTS/
22: C UNITS(4)=1002.
23: C
24: C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
25: C
26: C IF (UNICDE.EQ.0) UNICDE=-74
27: C
28: C SET CONVERSION FACTORS.
29: C
30: C DATA (FACTOR(1,1),1=1,3)/100,1.004,1.004/
31: C
32: C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
33: C
34: C UNITS(1)=UNITS
35: C UNITS(2)=ERROR
36: C
37: C
38: C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
39: C CODE INTO 'UNITS'.
40: C
41: C ICODE=IABS(UNICODE)
42: C DO 2 I=1,3
43: C IF (ICODE.NE.ENTRY(1,I)) GO TO 2
44: C IF (UNICODE.EQ.0) VALUE=VALUE*FACTOR(I)
45: C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
46: C UNITS(1)=ENTRY(2,I)
47: C UNITS(2)=ENTRY(3,I)
48: C
49: C AT THIS POINT, CONVERSION IS COMPLETE.
50: C
51: C RETURN
52: C
53: C ELSE, CHECK REST OF TABLE.
54: C

```

```

55.      2 CONTINUE
56.      C
57.      C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
58.      C
59.      PRINT 101,ICODE
60.      IGI FORMAT(' ERROR--UNIT CODE',I3,' NOT IN INTENSITY TABLE.')
```

```

61.      RETURN
62.      END
```



QFOR,SM N,LINEAR,LINEAR

```

1. SUBROUTINE LINEAR(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF LINEAR MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,12),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(12)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,12)/2,3,4,5,6,7,8,9,10,11,12,06/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1=2,3),1=1,12)/"MILLIM","ETERS",
18. C "CENTIM","ETERS", "METERS",
19. C "KILOM","ETERS", "INCHES",
20. C "FEET", "YARDS",
21. C "KILOM","ETERS", "NAUT", "MILES",
22. C "STAT", "MILES", "FATHOM", "S",
23. C "MICRON", "S",
24. C
25. C ENTER INTERNAL UNITS.
26. C
27. C UNITS(3)="METERS",
28. C UNITS(4)=
29. C
30. C IF UNICDE = 0, THEN SET CODE TO STANDARD UNITS.
31. C
32. C IF (UNICDE.EQ.0) UNICDE=-4
33. C
34. C SET CONVERSION FACTORS.
35. C
36. C DATA (FACTOR(1,1),1=1,12)/.00100,0.100,1.00,1.00,2.5400050-2,
37. C .30480060,914401921700,9.14401921702,
38. C 1.85203,1.609350,1.828803600,1.00-67
39. C
40. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
41. C CODE INTO 'UNITS'.
42. C
43. C ICODE=IABS(UNICDE)
44. C DO 2 I=1,12
45. C IF (ICODE.NE.ENTRY(1,I)) GO TO 2
46. C IF (UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
47. C IF (UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
48. C DO 1 J=1,2
49. C UNITS(J)=ENTRY(I,J,1)
50. C 1 CONTINUE
51. C
52. C AT THIS POINT, CONVERSION IS COMPLETE.
53. C
54. C RETURN

```

```

55. C
56. C ELSE, CHECK NEXT OF TABLE.
57. C
58. C 2 CONTINUE
59. C
60. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
61. C
62. C UNITS(1)='UNITS '
63. C UNITS(2)='ERROR '
64. C PRINT 101, CODE
65. C 101 FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN LINEAR TABLE. ')
66. C RETURN
67. C END

```



55. UNITS(1)=UNITS \*  
56. UNITS(2)=ERROR \*  
57. PRINT 101,ICODE  
58. 101 FORMAT(' ERROR--UNIT CUGL',13,' NOT IN LONG TIME TABLE.')

59. RETURN

60. END



GELT,L N,MAIN,MAIN

MAIN PROGRAM FOR BUILDING NAVDAB DATA BASE

3/26/74 GFA

IMPLICIT INTEGER (A-Z)

PARAMETER LCRUN = 50,LCPPV = 4165

COMMON/SECTG / ID(7),IDATE,NATSECLSTADN,LEAP,NOCNU,NOSTA,

LSTADC,LSTADA,LSTAGE

DIMENSION SECTA(28)

EQUIVALENCE (ID(1),SECTA(1))

COMMON /EXPMT/EXNO,PNITE,PCRU,ETILAT,ETILON,ETIDAT,EAFLAT,EAFLON,

EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,

EASTAT(2),EESTAT(2),EAFLG

COMMON /CRUISE/CRUNO,PNATC,CALAT,CALON,CAIDAT,CAFLAT,CAFLON,

CAFODAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,

CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),

CAFLG

COMMON /ASTATN/SANO,SANRUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,

SAFODAT,SASTAT(2),PNATAS,PENV5,SAFLG

COMMON /RUN /PORA,RILAT,RIMS,MFLAT,RFNS,RILON,RIEM,MFLON,RFEM,

RIMO,RIDAT,RIVR,RIITH,RIZON,RFMO,RFDAY,RFYR,RFTH,

RFZON,NAVCOB,DTASRC,SYSSRC,SYSSCR,SKCTYP,RCRTYP,

CLAS,DESCR,WAVIDR,WAVIDR,WAVIDR,WAVIDR,WAVIDR,WAVIDR,

SEASTA,MNDIR,MNDVEL,VELUNT,SMLDIR,SMLHT,

SHTUNT,SMLPRD,SPDUNT,WEATHR,BOTDPH,OPHUNT,BTHSLP,

BATHY,INFO(1),INSTAT(2),RUNNO

REAL WAVDIR,WAVIDR,WAVIDR,SEASTA,MNDIR,MNDVEL,SMLDIR,

SMLHT,SMLPRD,WEATHR,BOTDPH,BTHSLP

COMMON /CPV /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),

NDATS,NMORS(50),NDATA,VALRNG(2,30,50),DATA(30000)

DIMENSION ICON(4,30),IPART(30,30),IVLRNG(2,30,50),IEQV(3000)

EQUIVALENCE (CON(1,1),ICON(1,1)),IPAR(1,1),IPAR(1,1),

(VALRNG(1,1),IVLRNG(1,1),IEQV(1,1))

REAL CON,PAR,DATA,VALRNG

DATA BBI /'B B'/

```

55. DATA B62 /'BB'//
56. DATA B83 /'B'//
57. DATA B84 /'UBB'//
58. DIMENSION CARDIN(14),NAVID(7),STABFN(3000),TEMP(2)
59. DATA NAVID // NAVSEA UNDERWATER ACOUSTIC DATA BANK //
60. DATA CRDTYP //
61. DATA SEQ //
62. DATA BLNK //
63. C
64. DIMENSION UNITS(4)
65. C
66. CALL NTRAN (10,22)
67. C
68. READ(15,1) NOFILP,NOATE
69. 1 FORMAT(I)
70. C
71. 5 CALL DSKRED(I,1,SECIZ)
72. C
73. C
74. C
75. DO 10 I=1,7
76. IF (IIO(I) .NE. NAVID(I)) GO TO 20
77. 10 CONTINUE
78. GO TO 25
79. C
80. C
81. C
82. 20 WRITE(16,2) (IIO(I),I=1,7)
83. 21 FORMAT (//, 'WRONG DISK MOUNTED',/,1X,7A6)
84. STOP
85. C
86. C
87. C
88. 25 NRUNS = 0
89. C
90. CALL NTRAN (11,10) WHENIND 11
91. C
92. 27 CALL CRDRED (CARDIN,SEQ,CRDTYP)
93. C
94. C
95. C
96. IF (CRDTYP .NE. 'NE') GO TO 50
97. C
98. IF (NOSTA .EQ. 0 .AND. NOCRU .EQ. 0) GO TO 37
99. C
100. C
101. C
102. WRITE(16,32) NOCRU,NOSTA
103. 32 FORMAT (//, 'ERROR-- NE CARD ENCOUNTERED. NOCRU=',15, ' NOSTA=',15)
104. STOP
105. C
106. 37 DECODE(14,CARDIN) IEXPN,MNOCRU
107. 34 FORMAT (14,13,12)
108. C
109. IF (IEXP .NE. IEXPN) GO TO 45
110. C
111. C
WRITE ERROR. DUPLICATE EXP #'S

```

```

112. C      WRITE (6,40) LEXP,LEXPM
113.
114.      40 FORMAT(//,' ERROR-- TRYING TO ENTER SAME EXPERIMENT NO.,2(5)
115.      STOP
116. C
117. C      CHECK FOR NEW # OF CRUISES < 0
118. C
119. C      45 IF (NNOCRU .GT. 0) GO TO 49
120. C
121. C      WRITE ERROR
122. C
123. C      WRITE(6,47) NNOCRU
124. C      47 FORMAT(//,' ERROR-- NUMBER OF CRUISES INVALID.,15)
125. C      STOP
126. C
127. C      49 NNOCRU = NNOCRU
128. C      LEXP = LEXP
129. C
130. C      EXNO = LEXP
131. C      CALL EXPINT
132. C
133. C      CALL PSKEXP(DATA,NSEC)
134. C
135. C      CALL DSKRIT(NATSEC,NSEC,DATA)
136. C      PRINT 48
137. C      48 FORMAT (,' NEW EXP WRITTEN')
138. C
139. C      IF (LSTADR .EQ. 0) GO TO 46
140. C      FLD (18,18,SECADR) = FLD(18,18,LSTADR)
141. C      FLD (18,18,NSEC ) = FLD(18,18,LSTADR)
142. C
143. C      CALL DSKRED (SECADR,NSEC,DATA)
144. C
145. C      UPDATE POINTER
146. C
147. C      FLD(18,18,DATA(15)) = FLD(18,18,NATSEC)
148. C      FLD(18,18,DATA(15)) = FLD(18,18,NSEC)
149. C      CALL DSKRIT (SECADR,NSEC,DATA)
150. C
151. C      46 FLD(18,18,LSTADR) = FLD(18,18,NATSEC)
152. C      FLD(18,18,LSTADR) = FLD(18,18,NSEC)
153. C      NATSEC = NATSEC + NSEC
154. C
155. C      LSTADC = 0
156. C
157. C      READ NEXT CARD
158. C
159. C      GO TO 27
160. C
161. C      CHECK FOR NEW CRUISE CARD
162. C
163. C
164. C      50 IF (CROTP .NE. 'NC') GO TO 130
165. C
166. C      CHECK STATION COUNT FOR ZERO
167. C
168. C      55 IF (NOSTA .EQ. 0) GO TO 60

```

```

169. C
170. C WRITE ERROR
171. C
172. WRITE (6,57) LEXP,NOCRU,NOSTA,(CARDIN(1),1,1,14)
173. 57 FORMAT(' ERROR--NEW CRUISE ENCOUNTERED...',NOSTA NE 0',/,311J,/,
174. , 1A',13A6,A2)
175. STOP
176. C
177. 60 DECODE (61,CARDIN) IEXP,CRUNO,NNOSTA,NCARDS
178. 61 FORMAT(14,13,12,13,24,13)
179. IF (IEXP.EQ.LEXP) GO TO 70
180. C
181. C WRITE ERROR
182. C
183. WRITE (6,63) LEXP,(CARDIN(1),1,1,14)
184. 63 FORMAT(' ERROR-- EXPERIMENT NO. NOT SAME...',110,/,1A',13A6,A2)
185. STOP
186. C
187. C CHECK STATN NO .LT. 0
188. C
189. 70 IF (NNOSTA .GT. 0) GO TO 75
190. C
191. C WRITE ERROR
192. C
193. WRITE (6,72) NNOSTA,(CARDIN(1),1,1,14)
194. 72 FORMAT(' ERROR-- NEW NO. STATIONS INCORRECT...',110,/,1A',13A6,A2)
195. STOP
196. C
197. 75 NOSTA = NNOSTA
198. C
199. C READ IN ALL NOTE CARDS
200. C
201. C DO 90 I=1,NCARDS
202. READ (9,85) (NOTES(J),J=1,14)
203. 85 FORMAT (13A6,A2)
204. 90 CONTINUE
205. C
206. C CALL CRUINI
207. C
208. C CALL PKCRUIDATA,MSEC)
209. C
210. C CALL DSKRIT(INXTSEC,MSEC,DATA)
211. PRINT 95
212. 95 FORMAT (' NEW CRUISE WRITTEN')
213. C
214. C CHECK CRUISE # = 1
215. C
216. C IF (LSTADC .NE. 3) GO TO 110
217. C
218. C FLD(18,18,SECADR) = FLD(0,18,LSTADR)
219. FLD(18,18,MSEC) = FLD(18,18,LSTADR)
220. C
221. C CALL DSKRED (SECADR,MSEC,DATA)
222. C
223. C UPDATE DOWN PTR.
224. C
225. C

```



228.	FLD(0,18,DATA(16)) = FLD(18,18,NATSEC)
229.	FLD(18,18,DATA(16)) = FLD(18,18,NSEC)
230.	C
231.	C
232.	WRITE IT BACK
233.	CALL DSKRIT (SECADR,MSEC,DATA)
234.	C
235.	FLD(18,36,LSTADC) = FLD(0,36,DATA(16))
236.	C
237.	C
238.	ZERO LAST ADDRESSES FOR ACOUSTIC AND ENV STATIONS
239.	LSTADA = 0
240.	LSTADE = 0
241.	C
242.	NATSEC = NATSEC + NSEC
243.	C
244.	GO BACK & READ NEXT CARD
245.	C
246.	GO TO 27
247.	C
248.	J10 FLD(18,18,SECADR) = FLD(0,18,LSTADC)
249.	FLD(18,18,NSEC) = FLD(18,18,LSTADC)
250.	C
251.	CALL DSKREDISECADR,MSEC,DATA)
252.	C
253.	UPDATE NEXT PTR.
254.	C
255.	FLD(0,18,DATA(3)) = FLD(18,18,NATSEC)
256.	FLD(18,18,DATA(3)) = FLD(18,18,NSEC)
257.	C
258.	WRITE IT BACK
259.	C
260.	CALL DSKRIT (SECADR,MSEC,DATA)
261.	C
262.	UPDATE LAST CHUISE ADDRESS & SET NEXT AVAIL. ADDRESS
263.	FLD(18,36,LSTADC) = FLD(0,36,DATA(3))
264.	C
265.	NATSEC = NATSEC + NSEC
266.	C
267.	GO BACK & READ NEXT CARD
268.	C
269.	GO TO 27
270.	C
271.	----
272.	END OF PROCESSING FOR NE, NC & NT CARD TYPES ----
273.	C
274.	C
275.	CHECK FOR ACOUSTIC RUN ('A1')
276.	130 IF (CRDTYP .NE. 'A1') GO TO 176
277.	C
278.	C
279.	----
280.	PROCESS ACOUSTIC RUN INFO--
281.	C
282.	CHECK NOSTA.

A  
1  
00

```

283. IF INOSTA .NE. 0) GO TO 140
284. C
285. C WRITE ENRON
286. C
287. C
288. C WRITE (6,132) (CARDIN(1),1=1,14)
289. C 132 FORMAT (10 ERROR -- NEW ACOUSTIC RUN ENCOUNTERED. MUSTA * 0.0,/,1X,
290. C * 1346,A2)
291. C STOP
292. C
293. C CHECK FOR SEQ *10*
294. C
295. C 140 IF (SEQ,SEQ *10*) GO TO 146
296. C
297. C WRITE ERROR
298. C
299. C WRITE (6,145) (CARDIN(1),1=1,14)
300. C 145 FORMAT (10 ERROR -- ACOUSTIC RUN SEQ NOT 10.0,/,1X,1346,A2)
301. C STOP
302. C
303. C INCREMENT # OF RUNS COUNTER
304. C
305. C 146 NRUNS = NRUNS + 1
306. C
307. C DECODE DATA IN HEADER CARD (EXCEPT TIME)
308. C
309. C DECODE (147,CARDIN) PONA,MILAT,RINS,RFLAT,RFMS,RILON,RIEW,RFLON,
310. C RFWR,RIMO,RTDAY,RTYR,RIZON,RFM0,RFOAY,RFYR,
311. C RFZON,NAVCOU,DIASRC,SYSSRC,SYSKCH,SRCTYP,
312. C NCRTYP,CLAS,DESCR,RUNNO
313. C 147 FORMAT (114,A1,2(15,A1),2(16,A1),2(12,9A,A1),7(12,14,T11,13)
314. C
315. C EXAMINE TIME FIELD
316. C
317. C TIMST = BLNK
318. C CALL MOVINT (CARDIN,47,4)
319. C CALL MOVCHR (TIMST)
320. C
321. C IF (TIMST .NE. BLNK) GO TO 150
322. C
323. C WRITE ERROR
324. C
325. C WRITE (6,148) (CARDIN(1),1=1,14)
326. C 148 FORMAT (10 ERROR -- TIME FIELD BLANK.0,/,1X,1346,A2)
327. C STOP
328. C
329. C CHECK FOR *B* INDICATOR
330. C
331. C 150 RITIM = 4095
332. C IF (TIMST .EQ. B81) GO TO 155
333. C
334. C DECODE (153,TIMST) RITIM
335. C 152 FORMAT (14)
336. C
337. C DO OTHER TIME FIELD
338. C
339. C 155 TIMST = BLNK
340. C CALL MOVINT (CARDIN,58,4)

```

```

340. CALL MOUVR (TIMTST)
341. C
342. IF (TIMTST .NE. BLNK) GO TO 160
343. C
344. WRITE ERROR
345. C
346. WRITE (6,148) (CARDIN(1),1-1,14)
347. STOP
348. C
349. C CHECK FOR 'B'
350. C
351. 160 NRTIM = 1095
352. IF (TIMTST .EQ. B01) GO TO 162
353. C
354. DECODE (152,TIMTST) NRTIM
355. C
356. C
357. C READ A CARD & PROCESS 20,30,40,50 TYPES
358. C
359. 162 CALL CRORER (CARDIN,SEQ,CRDTYP)
360. NCON = 0
361. NPAR = 0
362. NVAR = 0
363. CALL COPAVA (CRDTYP,SEQ,CARDIN)
364. C
365. C
366. C
367. C
368. CALL NTRAN (11,1,LCRUN,PORA,LSTAT)
369. CALL NTRAN (11,22)
370. IF (LSTAT .GT. 0) GO TO 170
371. C
372. C WRITE ERROR
373. C
374. WRITE (6,165) LSTAT
375. 165 FORMAT (' ERROR --ARUN WRITE TO 11',110)
376. STOP
377. C
378. C NEXT COMMON 'CPV'
379. C
380. 170 CALL NTRAN (11,1,LCCPV,NCON,LSTAT)
381. CALL NTRAN (11,22)
382. IF (LSTAT .GT. 0) GO TO 178
383. C
384. C WRITE ERROR
385. C
386. WRITE (6,175) LSTAT
387. 175 FORMAT (' ERROR --ACPV WRITE TO 11',110)
388. STOP
389. C
390. C WRITE DATA OUT
391. C
392. 178 CALL NTRAN (11,1,NDATA,DATA(1),LSTAT)
393. CALL NTRAN (11,22)
394. IF (LSTAT .GT. 0) GO TO 179
395. C
396. C

```

```

397. C WRITE ERROR
398. WRITE (6,181) LSTAT,NDATA
399. 181 FORMAT (' ERROR -- DATA WRITE TO 11',2110)
400. STOP
401. C
402. C GO BACK & READ NEXT CARD
403. C
404. C 179 GO TO 27
405. C
406. C
407. C -- END OF PROCESSING FOR ACOUSTIC RUNS
408. C
409. C
410. C
411. C REMIND UNIT 11
412. C
413. C 176 CALL NTRAM (11,10)
414. C
415. C SAILAT = -90000
416. C SAILFLG = 0
417. C SAILDAT = 991232
418. C SAILTIM = 4095
419. C SAILPAT = 90000
420. C SAILFON = 179400
421. C SAILDAT = 0
422. C SAILTIM = -4095
423. C SASTAT(1) = 0
424. C SASTAT(2) = 0
425. C
426. C LSLTH = 0 @ INIT LAST LENGTH FOR DATA POINTERS
427. C
428. C ..... INIT IOTPTR TO PROPER WORD BEFORE DO LOOP .....
429. C IOTPTR = 11
430. C DO 230 I=1,NRUNS
431. C
432. C READ IN DATA FOR A RUN
433. C
434. C CALL NTRAM (11,2,LCCRUN,PORALLSTAT)
435. C CALL NTRAM (11,22)
436. C IF (LSTAT.GT. 0) GO TO 180
437. C
438. C WRITE (6,177) LSTAT,1,NRUNS
439. C 177 FORMAT (' ERROR -- ARUN READ FROM 11',3110)
440. C STOP
441. C
442. C 180 CALL NTRAM (11,2,LCCPV,NCON,LSTAT)
443. C CALL NTRAM (11,22)
444. C IF (LSTAT.GT. 0) GO TO 190
445. C
446. C WRITE (6,182) LSTAT
447. C 182 FORMAT (' ERROR -- ACPV READ FROM 11',110)
448. C STOP
449. C
450. C 190 CALL NTRAM (11,2,NDATA,DATA(1),LSTAT)
451. C CALL NTRAM (11,22)
452. C IF (LSTAT.GT. 0) GO TO 191
453. C

```



```

454. C WRITE ERROR
455. C
456. WRITE (6,192) LSTAT,NDATA
457. 192 FORMAT (10 ENRON -- DATA HEAD FROM 111,2110)
458. STOP
459. C CONVERT LAT, LONG, DATE TO PROPER FORM
460. C
461. 191 IF (RINS.EQ.'S') RILAT = -RILAT
462. IF (RINS.EQ.'S') RFLAT = -RFLAT
463. IF (RIEN.EQ.'E') RILON = -RILON
464. IF (RIEN.EQ.'E') RFLON = -RFLON
465. C
466. RIDAY = RIDAY + RIMO*100 + RIYR*10000
467. RFDAY = RFDAY + RFMO*100 + RFYR*10000
468. C
469. CALL ACSTAT (RSTAT)
470. C
471. IF (RILAT.GT. SAJLAT) SAJLAT = RILAT
472. IF (RFLAT.GT. SAILAT) SAILAT = RFLAT
473. IF (RILAT.LT. SAFLAT) SAFLAT = RILAT
474. IF (RFLAT.LT. SAFLON) SAFLON = RFLAT
475. C
476. IF (SAIFLG.NE. 0) GO TO 193
477. SAIFLG = 1
478. SAILON = EAST (RILON,RFLON)
479. SAFLON = WEST (RILON,RFLON)
480. GO TO 194
481. 193 SAILON = EAST (RILON,SAILON)
482. SAILON = EAST (RFLON,SAFLON)
483. SAFLON = WEST (RILON,SAFLON)
484. SAFLON = WEST (RFLON,SAFLON)
485. C
486. 194 IF (RIDAY.LT. SAIDAT) SAIDAT = RIDAY
487. IF (RFDAY.GT. SAFDAT) SAFDAT = RFDAY
488. C
489. C
490. 195 SASSTAT(1) = ORISASSTAT(1),RSTAT(1)
491. SASSTAT(2) = ORISASSTAT(2),RSTAT(2)
492. C
493. CALL MAXMIN
494. C
495. CALL ACMPRS (STABFR,IOTPTR,LNTH)
496. C
497. CHECK SUBSCRIPT OVERFLOW
498. C
499. IF (IOTPTR.LT. 3000) GO TO 197
500. WRITE (6,196) IOTPTR,I,NRUNS
501. 196 FORMAT (10 ENRON -- ACOUSTIC STATION TOO BIG!,3110)
502. STOP
503. C
504. 197 J= IOTPTR -LNTH + 9
505. FLD(C,18,STABFR(J)) = FLD(18,18,LSTLTH)
506. NDATA = NDATA + 1
507. TEMP(1) = NSCTRS(NDATA)
508. FLD(18,18,STABFR(J)) = FLD(18,18,TEMP(1))
509. LSTLTH = TEMP(1) + LSTLTH
510. C

```

FUNCTION NSCTRS COMPUTES THE # SCTRS

```

511.      230 CONTINUE
512.      C
513.      DECODE (232,CANDTN) SAND
514.      232 FORMAT (19,12)
515.      SANMUN = NRUNS
516.      CALL ASCMPS (STABFM,3)
517.      C
518.      IOTPTR = IOTPTR - 1
519.      NSEC = NSCTRS(IOTPTR)
520.      C
521.      FLD(18,18,STABFM(1)) = NSEC
522.      C
523.      ISTRT = 11
524.      SPN = NSEC * NATSEC
525.      DO 250 I=1,NRUNS
526.      FLD(10,18,STABFM(ISTRT+9)) = FLD(0,18,STABFM(ISTRT+9)) + SPN
527.      ISTRT = FLD(18,18,STABFM(ISTRT))
528.      250 CONTINUE
529.      C
530.      CALL DSKRIT (NATSEC,NSEC,STABFM(1))
531.      PRINT 252
532.      252 FORMAT (1, 'ACOUSTIC STATION WRITTEN')
533.      C
534.      IF (LSTADA * NE * 0) GO TO 280
535.      C
536.      FLD(18,18,SECADR) = FLD(10,18,LSTADC)
537.      FLD(18,18,MSEC) = FLD(18,18,LSTADC)
538.      C
539.      CALL DSKRED (SECADR,MSEC,STABFM)
540.      C
541.      C
542.      C
543.      FLD(10,18,STABFM(16)) = FLD(18,18,NATSEC)
544.      FLD(18,18,STABFM(16)) = FLD(18,18,MSEC)
545.      C
546.      C
547.      C
548.      C
549.      CALL DSKRIT (SECADR,MSEC,STABFM)
550.      PRINT 255
551.      255 FORMAT (1, 'LINK DOWN FROM CRUISE')
552.      C
553.      C
554.      C
555.      C
556.      C
557.      280 FLD(18,18,SECADR) = FLD(0,18,LSTADA)
558.      FLD(18,18,MSEC) = FLD(18,18,LSTADA)
559.      C
560.      CALL DSKRED (SECADR,MSEC,STABFM)
561.      C
562.      FLD(18,18,STABFM(9)) = FLD(18,18,NATSEC)
563.      FLD(18,18,STABFM(9)) = FLD(18,18,MSEC)
564.      C
565.      C
566.      C
567.      CALL DSKRIT (SECADR,MSEC,STABFM)

```

```

568. PRINT 283
569. 283 FORMAT (1* LINK FROM PREV. STATION*)
570. C
571. LSTADA = STABFRI9)
572. C
573. C SET NEXT AVAILABLE ADDRESS
574. C
575. 290 NATSEC = NATSEC + NSEC
576. C
577. C
578. C CONTINUE ON TO WRITE OUT RUN DATA
579. C
580. CALL NTRAN (11,10)
581. C
582. C DO LOOP TO HEAD 11 & WRITE DATA ONLY TO DISK
583. C
584. DO 310 1=1,NRUNS
585. C
586. C READ IN DATA
587. C
588. CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
589. CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
590. CALL NTRAN (11,22)
591. CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
592. CALL NTRAN (11,22)
593. C
594. IF (LSTAT .GT. 0) GO TO 300
595. C
596. C WRITE ERROR
597. C
598. C
599. WRITE (4,295) LSTAT
600. 295 FORMAT (1* ERROR -- READING CPV ON RUN ACOUSTIC DATA*,110)
601. STOP
602. C
603. C COMPUTE NUMBER OF SECTORS FOR JUST DATA -- SHOULD AGREE W/ PREV. LOOP
604. C
605. 300 NDATA = NDATA + 1
606. C
607. NSEC = NSCTRS(NDATA1)
608. FLD(10,18,VALRNG(2,30,50)) = 5
609. FLD(10,18,VALRNG(2,30,50)) = NSEC
610. C
611. CALL DSKRIT (NXTSEC,NSEC,VALRNG(2,30,50))
612. C
613. C SET NEXT AVAILABLE ADDRESS
614. C
615. NATSEC = NATSEC + NSEC
616. C
617. 310 CONTINUE
618. C
619. C ** THIS ENDS THE ACOUSTIC PROCESSING FOR ONE STATION **
620. C
621. C
622. C
623. C
624. C

```

# BEGIN PROCESSING OF ENVIRONMENTAL DATA

```

625. C
626. C
627. C SET NO. RUNS TO ZERO
628. C
629. C 37G NRUNS = 0
630. C
631. C REWIND UNIT 11
632. C
633. C CALL NTRAN (11,10)
634. C NCON = 0
635. C PRINT 372
636. C 372 FORMAT (' FINISHED WITH ACOUSTICS')
637. C
638. C CHECK FOR ENV. RUN CARD
639. C
640. C 38G IF (CRDTYP.EQ.'E1') GO TO 400
641. C
642. C WRITE ERROR
643. C
644. C
645. C WRITE (6,130) (CARDIN(1),1,1,14)
646. C 390 FORMAT (' ERROR -- ENVIRONMENTAL RUN CARD EXPECTED',/,1X,13A6,A2)
647. C STOP
648. C
649. C VERIFY SEQUENCE # '10'
650. C
651. C 400 IF (SEQ.EQ.'10') GO TO 415
652. C
653. C WRITE ERROR
654. C
655. C WRITE (6,1410) (CARDIN(1),1,1,14)
656. C 410 FORMAT (' ERROR -- EXPECTED SEQ 10 CARD',/,1X,13A6,A2)
657. C STOP
658. C
659. C INCREMENT RUN COUNTER & DECODE DATA (EXCEPT TIME)
660. C
661. C 415 NRUNS = NRUNS + 1
662. C NCON = 0
663. C NDATS = 0
664. C
665. C DECODE (420,CARDIN) FORA,R1LAT,MINS,RFLAT,RFNS,R1LON,R1GM,RFLON,
666. C RFEM,R1MO,R1DAY,R1YR,R1ZON,R1NO,R1DAY,R1YR,
667. C RFZON,NAVCOD,DYASRC,CLAS,DESCR,RUNNO
668. C 420 FORMAT (114,A1,2,15,A1),2(16,A1),2(312,4X,A1),2(12,8X,12,14,
669. C 711,13)
670. C
671. C EXAMINE TIME FIELD (THIS SECTION SAME AS ACOUSTICS)
672. C
673. C TIMST = BLNK
674. C CALL MOVINT (CARDIN,47,4)
675. C CALL MOVCHR (TIMST)
676. C
677. C IF (TIMST.NE.BLNK) GO TO 440
678. C
679. C WRITE ERROR
680. C
681. C WRITE (6,146) (CARDIN(1),1,1,14)

```



682.	STOP
683.	C
684.	C
685.	C
686.	440 RTIM = 4095
687.	IF (TIMST.EQ. 881) GO TO 463
688.	C
689.	DECODE (152,TIMST) RTIM
690.	C
691.	DO OTHER TIME FIELD
692.	C
693.	440 TIMST = BLNK
694.	CALL MOVINT (CARDIN,58,4)
695.	CALL MOVCHR (TIMST)
696.	C
697.	IF (TIMST.NE. BLNK) GO TO 480
698.	C
699.	WRITE ERROR
700.	C
701.	WRITE (6,148) (CARDIN(1),1,14)
702.	STOP
703.	C
704.	C
705.	C
706.	480 RTIM = 4095
707.	IF (TIMST.EQ. 881) GO TO 500
708.	C
709.	DECODE (152,TIMST) RTIM
710.	C
711.	C
712.	C
713.	READ NEXT CARD
714.	500 CALL CROED (CARDIN,SEQ,CROTP)
715.	C
716.	C
717.	PROCESS '11' TYPE CARD
718.	C
719.	IF (SEQ.EQ. '11') GO TO 520
720.	C
721.	WRITE ERROR
722.	C
723.	WRITE (6,510) (CARDIN(1),1,14)
724.	510 FORMAT (1' ERROR -- EXPECTED SEQ 11 CARD.',/,1X,13A6,A2)
725.	STOP
726.	C
727.	C
728.	CONTINUE TO DECODE CARD
729.	C
730.	520 DECODE (525,CARDIN) #AVMT,HTUNIT,#HTMTD,#WVPRD,PRDUNT,PRDMD,
731.	#NDIR,DIRUNT,DIRMTD,#NDVEL,VELUNT,VELMD,
732.	#LHFT,SHUNT,SHMTD,#SLPRD,SPDUNT,SPDMD
733.	525 FORMAT (16,2(F3.0,213),1X,F3.0,213,F4.0,213,2X,
734.	2(F3.0,213))
735.	C
736.	C
737.	C
738.	C
739.	C
740.	C
741.	C
742.	C
743.	C
744.	C
745.	C
746.	C
747.	C
748.	C
749.	C
750.	C
751.	C
752.	C
753.	C
754.	C
755.	C
756.	C
757.	C
758.	C
759.	C
760.	C
761.	C
762.	C
763.	C
764.	C
765.	C
766.	C
767.	C
768.	C
769.	C
770.	C
771.	C
772.	C
773.	C
774.	C
775.	C
776.	C
777.	C
778.	C
779.	C
780.	C
781.	C
782.	C
783.	C
784.	C
785.	C
786.	C
787.	C
788.	C
789.	C
790.	C
791.	C
792.	C
793.	C
794.	C
795.	C
796.	C
797.	C
798.	C
799.	C
800.	C
801.	C
802.	C
803.	C
804.	C
805.	C
806.	C
807.	C
808.	C
809.	C
810.	C
811.	C
812.	C
813.	C
814.	C
815.	C
816.	C
817.	C
818.	C
819.	C
820.	C
821.	C
822.	C
823.	C
824.	C
825.	C
826.	C
827.	C
828.	C
829.	C
830.	C
831.	C
832.	C
833.	C
834.	C
835.	C
836.	C
837.	C
838.	C
839.	C
840.	C
841.	C
842.	C
843.	C
844.	C
845.	C
846.	C
847.	C
848.	C
849.	C
850.	C
851.	C
852.	C
853.	C
854.	C
855.	C
856.	C
857.	C
858.	C
859.	C
860.	C
861.	C
862.	C
863.	C
864.	C
865.	C
866.	C
867.	C
868.	C
869.	C
870.	C
871.	C
872.	C
873.	C
874.	C
875.	C
876.	C
877.	C
878.	C
879.	C
880.	C
881.	C
882.	C
883.	C
884.	C
885.	C
886.	C
887.	C
888.	C
889.	C
890.	C
891.	C
892.	C
893.	C
894.	C
895.	C
896.	C
897.	C
898.	C
899.	C
900.	C
901.	C
902.	C
903.	C
904.	C
905.	C
906.	C
907.	C
908.	C
909.	C
910.	C
911.	C
912.	C
913.	C
914.	C
915.	C
916.	C
917.	C
918.	C
919.	C
920.	C
921.	C
922.	C
923.	C
924.	C
925.	C
926.	C
927.	C
928.	C
929.	C
930.	C
931.	C
932.	C
933.	C
934.	C
935.	C
936.	C
937.	C
938.	C
939.	C
940.	C
941.	C
942.	C
943.	C
944.	C
945.	C
946.	C
947.	C
948.	C
949.	C
950.	C
951.	C
952.	C
953.	C
954.	C
955.	C
956.	C
957.	C
958.	C
959.	C
960.	C
961.	C
962.	C
963.	C
964.	C
965.	C
966.	C
967.	C
968.	C
969.	C
970.	C
971.	C
972.	C
973.	C
974.	C
975.	C
976.	C
977.	C
978.	C
979.	C
980.	C
981.	C
982.	C
983.	C
984.	C
985.	C
986.	C
987.	C
988.	C
989.	C
990.	C
991.	C
992.	C
993.	C
994.	C
995.	C
996.	C
997.	C
998.	C
999.	C
1000.	C

```

739. IF (SHTUNT.EQ.0.AND. SWELLT.NE.0) GO TO 530
740. IF (SPDUNT.EQ.0.AND. SWLPRD.NE.0) GO TO 530
741. GO TO 540
742. C
743. C WRITE ERROR
744. C
745. S33 WRITE (6,535) (CARDIN(1),1,14)
746. S35 FORMAT (1, ERROR -- UNIT FIELD = J, DATA FIELD NOT.,1X,13A6,A2)
747. STOP
748. C
749. C CLEAR STATUS WORDS
750. C
751. S40 RSTAT(1) = 0
752. RSTAT(2) = J
753. C
754. C CONVERT UNITS & MOVE DATA TO CONSTANTS AREA
755. C
756. C WAVE HEIGHT
757. C
758. IF (HTUNIT.EQ.0) GO TO 550
759. CALL STOR(102,HWHT,HTUNIT,SHMTD)
760. C
761. C WAVE PERIOD
762. C
763. S50 IF (PRDUNT.EQ.0) GO TO 552
764. CALL STOR(103,WPVRD,PRDUNT,PRDMDT)
765. C
766. C WIND DIRECTION
767. C
768. S52 IF (DIRUNT.EQ.0) GO TO 555
769. CALL STOR(105,WDDIR,DIRUNT,DIRMDT)
770. C
771. C WIND VELOCITY
772. C
773. S55 IF (VELUNT.EQ.0) GO TO 560
774. CALL STOR(106,WVVEL,VELUNT,VELMDT)
775. C
776. C SWELL HEIGHT
777. C
778. S60 IF (SHTUNT.EQ.0) GO TO 565
779. CALL STOR(108,SWELLT,SHTUNT,SHMTD)
780. C
781. C SWELL PERIOD
782. C
783. S65 IF (SPDUNT.EQ.0) GO TO 600
784. CALL STOR(109,SWLPRD,SPDUNT,SPDMDT)
785. C
786. C PROCESS WAVE DIRECTION FIELD
787. C
788. S60 CALL MOVINT (CARDIN,14,2)
789. TEMP11 = BLNK
790. CALL MOVCHR (TEMP)
791. IF (TEMP11.EQ. BB2) GO TO 620
792. IF (TEMP11.NE. BLNK) GO TO 610
793. WRITE (6,635) (CARDIN(1),1,14)
794. S65 FORMAT (1, ERROR -- WAVE DIRECTION FIELD BLANK.,1X,13A6,A2)
795. STOP

```

```

796*      617 DECODE (611,TEMP) AAVDIR
797*      611 FORMAT(F2.0)
798*      CALL STOR (101,NAVDIR,1.0)
799*      C
800*      C      PROCESS SEA STATE FIELD
801*      C
802*      620 CALL MOVINT (CARDIN,34,1)
803*      TEMP(1) = BLNK
804*      CALL MOVCHR (TEMP)
805*      IF (TEMP(1) .EQ. 883) GO TO 680
806*      IF (TEMP(1) .NE. BLNK) GO TO 630
807*      WRITE (6,625) (CARDIN(1),1,1,14)
808*      625 FORMAT(1X,625) (CARDIN(1),1,1,14)
809*      STOP
810*      630 DECODE (631,TEMP) SEASTA
811*      631 FORMAT (F1.0)
812*      CALL STOR (104,SEASTA,1.0)
813*      C
814*      C      PROCESS SWELL DIRECTION FIELD
815*      C
816*      680 CALL MOVINT (CARDIN,54,2)
817*      CALL MOVCHR (TEMP)
818*      IF (TEMP(1) .EQ. 882) GO TO 700
819*      IF (TEMP(1) .NE. BLNK) GO TO 690
820*      WRITE (6,685) (CARDIN(1),1,1,14)
821*      685 FORMAT (1X,685) (CARDIN(1),1,1,14)
822*      STOP
823*      690 DECODE (611,TEMP) SWLOIN
824*      CALL STOR (107,SWLOIN,1.0)
825*      C
826*      C      PROCESS WEATHER CODE FIELD
827*      C
828*      700 CALL MOVINT (CARDIN,74,2)
829*      CALL MOVCHR (TEMP)
830*      IF (TEMP(1) .EQ. 882) GO TO 720
831*      IF (TEMP(1) .NE. BLNK) GO TO 710
832*      WRITE (6,705) (CARDIN(1),1,1,14)
833*      705 FORMAT (1X,705) (CARDIN(1),1,1,14)
834*      STOP
835*      710 DECODE (611,TEMP) WEATHR
836*      CALL STOR (110,WEATHR,1.0)
837*      C
838*      C      PROCESS BOTTOM SLOPE FIELD
839*      C
840*      720 CALL MOVINT (CARDIN,76,3)
841*      TEMP(1) = BLNK
842*      CALL MOVCHR (TEMP)
843*      BTMSLP = 0
844*      IF (TEMP(1) .EQ. 884) GO TO 740
845*      IF (TEMP(1) .NE. BLNK) GO TO 730
846*      WRITE (6,725) (CARDIN(1),1,1,14)
847*      725 FORMAT (1X,725) (CARDIN(1),1,1,14)
848*      STOP
849*      730 CALL MOVINT (CARDIN,76,1)
850*      TEMP(1) = BLNK
851*      CALL MOVCHR (TEMP)
852*      IF (TEMP(1) .NE. BLNK) BTMSLP = 1

```

```

853. CALL MOVCHR (TEMP)
854. IF (TEMP(1) .EQ. BLNK) GO TO 735
855. IF (BTMSLP .EQ. 0.) GO TO 733
856. WRITE (6,731) (CARDIN(1),1,14)
857. 731 FORMAT (1, ERROR -- MULTIPLE CHECKS IN BOTTOM SLOPE FIELD.,/,1X,
858. , 13A6,A2)
859. STOP
860. 733 BTMSLP = 2.
861. 735 CALL MOVCHR (TEMP)
862. IF (TEMP(1) .EQ. BLNK) GO TO 740
863. IF (BTMSLP .EQ. 0.) GO TO 737
864. WRITE (6,731) (CARDIN(1),1,14)
865. STOP
866. 737 BTMSLP = 3.
867. 740 CALL STOR (111,BTMSLP,1,0)
868. C
869. C PROCESS BATHOMETRY FIELD
870. C
871. 743 CALL MOVINT (CARDIN,79,1)
872. TEMP(1) = BLNK
873. CALL MOVCHR (TEMP)
874. IF (TEMP(1) .NE. 1) GO TO 741
875. RSTAT(2) = RSTAT(2) + 4
876. GO TO 750
877. 741 IF (TEMP(1) .EQ. 'N') GO TO 750
878. WRITE (6,742) (CARDIN(1),1,14)
879. 742 FORMAT (1, ERROR -- BATHOMETRY FIELD NOT Y OR N.,/,1X,13A6,A2)
880. STOP
881. C
882. C -- END OF PROCESSING ERI1 CARD --
883. C
884. C
885. C READ NEXT CARD
886. C
887. 750 CALL CROED (CARDIN,SEQ,CRODTP)
888. C
889. C VERIFY SEQ = '12'
890. C
891. IF (SEQ .EQ. '12') GO TO 760
892. C
893. C WRITE ERROR
894. C
895. WRITE (6,755) (CARDIN(1),1,14)
896. 755 FORMAT (1, ERROR -- EXPECTED SEQUENCE 12 CARD.,/,1X,13A6,A2)
897. STOP
898. 760 DECODE(761,CARDIN) BOTDPH,DPHUNT,DPHMTD
899. 761 FORMAT (14,F7.0,2I3)
900. IF (DPHUNT .EQ. 0. AND .BOTDPH .NE. 0) GO TO 530
901. IF (DPHUNT .EQ. 0) GO TO 7615
902. CALL STOR(113,BOTDPH,DPHUNT,DPHMTD)
903. C
904. C MOVE DATA FIELD
905. C
906. 7615 CALL MOVINT (CARDIN,27,54)
907. CALL MOVCHR (INFO)
908. C
909. C CLEAR END OF FILE FLAG

```



```

913. C      EOF = 0
914. C
915. C      READ NEXT CARD & CHECK END OF FILE
916. C
917. C      READ(9,762,END=765) (CARDIN(1),1=1,14)
918. C      FORMAT (13A6,A2)
919. C      IF (FLO(0,6,CARDIN(1)) .EQ. FLO(0,6,0)) GO TO 765
920. C      FLD(0,12,SEQ) = FLD(0,12,CARDIN(1))
921. C      GO TO 770
922. C
923. C      SET EOF FLAG & CLEAN SEQ
924. C
925. C      765 EOF = 1
926. C      SEQ = '00'
927. C
928. C      770 NPAR = Q
929. C      NVAR = Q
930. C
931. C      CHECK NEW CARD FOR 20, 30, 40 TYPE
932. C
933. C      IF (.NOT. (SEQ .EQ. '20' .OR. SEQ .EQ. '30' .OR. SEQ .EQ. '40'))
934. C      *      GO TO 775
935. C
936. C      PROCESS 'CPV' TYPE CARDS
937. C
938. C      FLD(0,12,CROTP) = FLD(0,12,CARDIN(1))
939. C
940. C      CALL COPAVA (CROTP,SEQ,CARDIN)
941. C
942. C      READ NEXT CARD & CHECK FOR EOF
943. C
944. C      READ(9,762,END=780) (CARDIN(1),1=1,14)
945. C      IF (FLO(0,6,CARDIN(1)) .EQ. FLO(0,6,0)) GO TO 780
946. C      FLD(0,12,CROTP) = FLD(0,12,CARDIN(1))
947. C      FLD(0,12,SEQ) = FLD(0,12,CARDIN(1))
948. C      GO TO 800
949. C
950. C      775 ' (NCON .EQ. 0) GO TO 800
951. C      DO 778 I=1,NCON
952. C      CALL CONVERT(CON(I),CON(I),CON(2,1),UNITS)
953. C      IF (UNITS(I) .EQ. 'UNITS') WRITE(6,777) (CON(I,1),J=1,3)
954. C      777 FORMAT (' CONSTANTS ',2I5,F21.7)
955. C      778 CONTINUE
956. C      GO TO 800
957. C
958. C      SET EOF FLAG
959. C
960. C      780 EOF = 1
961. C
962. C      DO COMMON 'RUN'
963. C
964. C      800 CALL NTRAN (1,1,LCRUN,PORA,LSIAT)
965. C      CALL NTRAN (1,22)
966. C      IF (LSIAT .GT. 0) GO TO 810
967. C

```

```

967. C      WRITE EYRON
968. C
969.      WRITE (6,805) LSTAT
970.      BUS FORMAT (1, ERROR) -- ENUN ANITE TO 11',110)
971.      STOP
972. C
973. C      HEAT COMMON CPU
974. C
975.      610 CALL NTRAN (11,1,LCCPV,NCUN,LSTAT)
976.      CALL NTRAN (11,22)
977.      IF (LSTAT .GT. 0) GO TO 820
978. C
979. C      WRITE ERROR
980. C
981.      WRITE (6,815) LSTAT
982.      BUS FORMAT (1, ERROR) -- ECPV ANITE TO 11',110)
983.      STOP
984. C
985. C      WRITE OUT DATA
986. C
987.      820 CALL NTRAN (11,1,NDATA,DATA(1),LSTAT)
988.      CALL NTRAN (11,22)
989.      IF (LSTAT .GE. 0) GO TO 821
990. C
991. C      WRITE ERROR
992. C
993.      WRITE (6,825) LSTAT
994.      BUS FORMAT (1, ERROR) -- EDATA ANITE TO 11',110)
995.      STOP
996. C
997. C      CHECK EOF FLAG. IF NOT SET, GO PROCESS CARD JUST READ
998. C
999.      821 IF (EOF .EQ. 0) GO TO 380
1000. C
1001. C      -- END OF PROCESSING FOR ENVIRONMENTAL RUNS
1002. C
1003. C
1004. C      REMIND UNIT 11
1005. C
1006. C
1007.      CALL NTRAN (11,10)
1008. C
1009. C      INIT FOR STATION SUMMARY
1010. C
1011.      SAILAT = -90000
1012.      SAILFG = 0
1013.      SAI DAT = 991232
1014.      SAITIM = 4095
1015.      SAFLAT = 90000
1016.      SAFLOM = 179600
1017.      SAFDAT = 0
1018.      SAFTIM = -4095
1019.      SASAT(1) = 0
1020.      SASAT(2) = 0
1021. C
1022. C      LSLIM = 0      @ INIT LAST LENGTH FOR RUN DATA POINTERS
1023. C

```

```

1221. C      --DO LOOP TO READ IN ENVIRONMENTAL RUNS
1225. C
1226. C      TOTPTR = 11
1227. C      DO 1000 J=1,NRUNS
1228. C
1229. C      READ IN DATA FOR A RUN
1230. C
1231. C      CALL NTRAN (11,2,LCNUN,PORA,LSTAT)
1232. C      CALL NTRAN (11,22)
1233. C      IF (LSTAT.GT. 0) GO TO 880
1234. C
1235. C      WRITE (6,877) LSTAT
1236. C      877 FORMAT (' ERROR -- ERUN READ FROM 11',110)
1237. C      STOP
1238. C
1239. C      880 CALL NTRAN (11,2,LCPPV,NCON,LSTAT)
1240. C      CALL NTRAN (11,22)
1241. C      IF (LSTAT.GT. 0) GO TO 890
1242. C
1243. C      WRITE (6,882) LSTAT
1244. C      882 FORMAT (' ERROR -- ECPV READ FROM 11',110)
1245. C      STOP
1246. C
1247. C      890 CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
1248. C      CALL NTRAN (11,22)
1249. C      IF (LSTAT.GE. 0) GO TO 891
1250. C
1251. C      WRITE ERROR
1252. C
1253. C      891 IF (IRINS.EQ. 'S') RILAT = -RILAT
1254. C      IF (IRFNS.EQ. 'S') RFLAT = -RFLAT
1255. C      IF (IRIEW.EQ. 'M') RILON = -RILON
1256. C      IF (RFEW.EQ. 'M') RFLOM = -RFLOM
1257. C      RIDAY = RIDAY + RIMO*100 + RIVM*10000
1258. C      MFDAY = MFDAY + RFMO*100 + RFRM*10000
1259. C
1260. C      CALL ENSTAT (RSTAT)
1261. C
1262. C      892 FORMAT (' ERROR -- EDATA READ FROM 11',110)
1263. C      STOP
1264. C
1265. C      CONVERT LAT, LONG, DATE TO PROPER FORM
1266. C
1267. C      893 IF (RILAT.EQ. 'S') RILAT = -RILAT
1268. C      IF (RFLAT.EQ. 'S') RFLAT = -RFLAT
1269. C      IF (RILON.EQ. 'M') RILON = -RILON
1270. C      IF (RFLOM.EQ. 'M') RFLOM = -RFLOM
1271. C      RIDAY = RIDAY + RIMO*100 + RIVM*10000
1272. C      MFDAY = MFDAY + RFMO*100 + RFRM*10000
1273. C
1274. C      CALL ENSTAT (RSTAT)
1275. C
1276. C      UPDATE STATION SUMMARY
1277. C
1278. C      IF (RILAT.GT. SAILAT) SAILAT = RILAT
1279. C      IF (RFLAT.GT. SAILAT) SAILAT = RFLAT
1280. C      IF (RILON.GT. SAFLAT) SAFLAT = RILON
1281. C      IF (RFLOM.GT. SAFLAT) SAFLAT = RFLOM
1282. C
1283. C      IF (SAILFLG.EQ. 'C') GO TO 893
1284. C      SAILFLG = 'I'
1285. C      SAILON = EAST(RILON,RFLOM)
1286. C      SAFLON = WEST(RILON,RFLOM)
1287. C      GO TO 894
1288. C

```

```

1081.      893 SAILON = EASTIRLON,SAILON)
1082.      SAILON = EASTIRFLON,SAILON)
1083.      SAILON = WESTIRFLON,SAILON)
1084.      SAILON = WESTIRFLON,SAILON)
1085.      C
1086.      894 IF (IRIDAY .LT. SAIDAT) SAIDAT = IRIDAY
1087.      IF (IRFDAY .GT. SAFOAT) SAFOAT = IRFDAY
1088.      C
1089.      C
1090.      895 SASSTAT(1) = OR(SASSTAT(1),RSTAT(1))
1091.      SASSTAT(2) = OR(SASSTAT(2),RSTAT(2))
1092.      C
1093.      CALL MAXMIN.
1094.      C
1095.      C      CALL ROUTINE TO COMPRESS DATA INTO OUTPUT ARRAY
1096.      C
1097.      CALL ACMPRS (STABFR,IOTPTR,LNTH)
1098.      C
1099.      CHECK SUBSCRIPT OVERFLOW
1100.      C
1101.      IF (IOTPTR .LT. 1000) GO TO 897
1102.      WRITE (5,896) IOTPTR,INKUNS
1103.      896 FORMAT(' ERROR -- ENV STATION TOO BIG(1',J11C)
1104.      STOP
1105.      C
1106.      897 J = IOTPTR - LNTH + 9
1107.      FLD(10,18,STABFR(J)) = FLD(10,18,LSLTH)
1108.      NDATA1 = NDATA + 10 * NDATA + 1 * NO. WORDS IN INFO FILED.
1109.      TEMP(1) = NSCTRS(NDATA1)
1110.      FLD(10,18,STABFR(J)) = FLD(10,18,TEMP(1))
1111.      LSLTH = TEMP(1) * LSLTH
1112.      C
1113.      1000 CONTINUE
1114.      C
1115.      SANRUN = NRUNS
1116.      CALL ASCHPS (STABFR,4)
1117.      C
1118.      IOTPTR = IOTPTR - 1
1119.      NSEC = NSCTRS (IOTPTR)
1120.      C
1121.      FLD(10,18,STABFR(1)) = NSEC
1122.      C
1123.      ISTAT = 1)
1124.      SPN = NSEC * NITSEC
1125.      DO 1050 I=1,NRUNS
1126.      FLD(10,18,STABFR(ISTAT+9)) = FLD(10,18,STABFR(ISTAT+9)) + SPN
1127.      ISTAT = FLD(10,18,STABFR(ISTAT))
1128.      1050 CONTINUE
1129.      C
1130.      PRINT ICS3
1131.      1053 FORMAT (' WRITE ENV STATION')
1132.      CALL DSKHIT INATSEC,NSEC,STABFR)
1133.      C
1134.      IF (LSTADE .NE. 0) GO TO 1040
1135.      C
1136.      FLD(10,18,SECADM) = FLD(10,18,LSTADC)
1137.      FLD(10,18,MSEC) = FLD(10,18,LSTADC)

```



```

1138. C      CALL DSKRED (SECADR,MSEC,STABFR)
1139. C
1140. C
1141. C      UPDATE DOWN ENV. PUT=10
1142. C
1143. C      FLD(0,18,STABFR(17)) = FLD(18,18,NATSEC)
1144. C      FLD(18,18,STABFR(17)) = FLD(18,18,NSEC)
1145. C
1146. C      WRITE IT BACK
1147. C
1148. C      CALL DSKRIT (SECADR,MSEC,STABFR)
1149. C      PRINT 1057
1150. C      1057 FORMAT (' LINK DOWN ENV')
1151. C
1152. C      LSTADE = STABFR(17)
1153. C
1154. C
1155. C      GO TO 1090
1156. C
1157. C      1080 FLD(18,18,SECADR) = FLD(0,18,LSTADE)
1158. C      FLD(18,18,MSEC) = FLD(18,18,LSTADE)
1159. C
1160. C      CALL DSKRED (SECADR,MSEC,STABFR)
1161. C
1162. C      UPDATE PTR TO NEXT STATION
1163. C
1164. C
1165. C      FLD(0,18,STABFR(9)) = FLD(18,18,NATSEC)
1166. C      FLD(18,18,STABFR(9)) = FLD(18,18,NSEC)
1167. C
1168. C      WRITE ENV STATION BACK OUT
1169. C
1170. C      CALL DSKRIT (SECADR,MSEC,STABFR)
1171. C      PRINT 1083
1172. C      1083 FORMAT (' LINK FROM ENV')
1173. C
1174. C      LSTADE = STABFR(9)
1175. C
1176. C      SET NEXT AVAILABLE ADDRESS
1177. C      1090 NATSEC = NATSEC + NSEC
1178. C
1179. C
1180. C      CONTINUE ON TO WRITE RUN DATA TO DISK
1181. C
1182. C      CALL NTRAN (11,10)
1183. C
1184. C      DO LOOP TO READ II & WRITE SHIP NAME DATA & VARIABLE DATA TO
1185. C      DISK
1186. C
1187. C      DO 1110 I=1,NRUNS
1188. C
1189. C      READ IN DATA
1190. C
1191. C      CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
1192. C      CALL NTRAN (11,2,LCCPV,NCON,LSTAT)
1193. C      CALL NTRAN (11,22)
1194. C      CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)

```

```

1195.      CALL NTRAN (11,22)
1196.      C
1197.      C
1198.      IF (LSTAT .GE. 0) GO TO 1100
1199.      C
1200.      C      WRITE ERROR
1201.      C
1202.      WRITE (6,1095) LSTAT
1203.      1095 FORMAT (1,ERROR -- READING CPV OR RUN ENV DATA',11J)
1204.      STOP
1205.      C
1206.      C      MOVE SHIP NAME DATA TO LAST 9 WORDS IN 'CPV' BEFORE 'DATA'.
1207.      C
1208.      1100 DO 1105 J=1,9
1209.      1EQV(2991+J) = INFO(J)
1210.      1105 CONTINUE
1211.      C
1212.      C      COMPUTE NO. OF SECTIONS TO WRITE OUT
1213.      C
1214.      TEMP(1) = NDATA + 10
1215.      NSEC = NSCTRS(TEMP(1))
1216.      C
1217.      FLD( 0,18,1EQV(2991)) = 6
1218.      FLD(18,18,1EQV(2991)) = NSEC
1219.      C
1220.      C
1221.      CALL DSKRIT (NXTSEC,NSEC,1EQV(2991))
1222.      C
1223.      C
1224.      HXTSEC = NXTSEC + NSEC
1225.      C
1226.      1110 CONTINUE
1227.      C
1228.      C
1229.      C
1230.      C
1231.      FLD(18,18,SECADR) = FLD( 0,18,LSTADA)
1232.      FLD(18,18,MSEC) = FLD(18,18,LSTADA)
1233.      C
1234.      C      READ IN ACOUSTIC STATION
1235.      C
1236.      CALL DSKRED (SECADR,MSEC,STABFR)
1237.      C
1238.      STABFR(10) = LSTADE
1239.      C
1240.      C      WRITE ACOUSTIC STATION BACK
1241.      C
1242.      C
1243.      C      CALL DSKRIT (SECADR,MSEC,STABFR)
1244.      C
1245.      FLD(18,18,SECADR) = FLD( 0,18,LSTADE)
1246.      FLD(18,18,MSEC) = FLD(18,18,LSTADE)
1247.      C
1248.      C      READ IN ENVIRONMENTAL STATION
1249.      C
1250.      CALL DSKRED (SECADR,MSEC,STABFR)
1251.      C
1252.      STABFR(10) = LSTADA

```

```

1252. C
1253. C WRITE ENV STATION BACK
1254. C
1255. C CALL DSKRIT (SECAUR,MSEC,STABFM)
1256. C
1257. C CALL CRUSUM (DATA,STABFM)
1258. C PRINT 1150
1259. C 1150 FORMAT (' LAST CRUSUM ')
1260. C
1261. C CALL EXPSUM (DATA,STABFM)
1262. C PRINT 1159
1263. C 1159 FORMAT (' PAST EXPSUM ')
1264. C
1265. C DECREMENT STATION COUNTER BY 1
1266. C
1267. C NOSTA = NOSTA -1
1268. C
1269. C IF THAT WAS LAST STATION DECREMENT CRUISE COUNTER
1270. C
1271. C IF (NOSTA .NE. 0) GO TO 1175
1272. C NOCRU = NOCRU -1
1273. C LSTADA = 0
1274. C LSTADE = 0
1275. C
1276. C UPDATE HEADER & WRITE IT BACK
1277. C
1278. C 1175 IDATE = NDATE
1279. C CALL DSKRIT (Q,I,SECTZ)
1280. C
1281. C DECREMENT NO. OF FILES PROCESSED
1282. C
1283. C NOFILP = NOFILP -1
1284. C
1285. C IF NOT FINISHED, DO NEXT FILE
1286. C
1287. C WRITE (6,1200)
1288. C 1200 FORMAT (' *** ANOTHER STATION COMPLETED *** ')
1289. C
1290. C IF (NOFILP .NE. 0) GO TO 5
1291. C STOP
1292. C END
1293.

```

QELT,L N,MAXMIN,,MAXMIN

SUBROUTINE MAXMIN

```
1. C
2. C MAXMIN (MAXIMUM-MINIMUM) COMPUTES THE MAX & MIN VALUES FOR
3. C EACH VARIABLE IN EACH DATA SET IN 'DATA' OF COMMON /CPV/.
4. C THE MAX MIN VALUES ARE STORED IN 'VALRG' IN COMMON /CPV/.
5. C
6. C
7. C THIS PROGRAM WAS CHALKBOARD 'DEBUGGED' BY DENNIS BOWEN
8. C AND JEFF ANDERSON ON 22 MAY 1974. GFA
9. C
10. C IMPLICIT INTEGER (A-Z)
11. C
12. C
13. C COMMON /CPV /NCON,CON(1,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
14. C NDATS,NROWS(50),NDATA,VALRG(2,30,50),DATA(30000)
15. C DIMENSION ICON(1,30),IPAR(30,30),IVLNG(2,30,50)
16. C EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
17. C (VALRG(1,1),IVLNG(1,1))
18. C REAL CON,PAR,DATA,VALRG
19. C
20. C
21. C REAL AMAX,AMIN
22. C DATA ANULL /0777777777776/
23. C
24. C L = 1
25. C
26. C DO 30 J=1,NDATS
27. C N = NROWS(J) * NVAR + L - 1
28. C M = NVAR + L - 1
29. C
30. C DO 20 K=L,M
31. C KK = K-L+1
32. C AMIN = 10E37
33. C AMAX = -10E37
34. C
35. C DO 10 I=K,M,NVAR
36. C IF (DATA(I),EQ,ANULL) GO TO 10
37. C IF (DATA(I) .LT. AMIN) AMIN = DATA(I)
38. C IF (DATA(I) .GT. AMAX) AMAX = DATA(I)
39. C 10 CONTINUE
40. C
41. C VALRG(1,KK,J) = AMIN
42. C VALRG(2,KK,J) = AMAX
43. C 20 CONTINUE
44. C
45. C L = N+1
46. C 30 CONTINUE
47. C
48. C
49. C RETURN
50. C END
```



RELT,L 4,MOVINT,MOVINT

```

1. SUBROUTINE MOVINT(FROM,ISTRT,NOCHR)
2. C
3. C MOVINT (MOVE INITIALIZE) INITIALIZES VARIABLES FOR FUTURE
4. C CALLS TO MOVCHR.
5. C
6. C FROM - ADDRESS OF FIELD CHARACTERS ARE TO BE MOVED
7. C FROM (STORED 6 CHAR. PER WORD)
8. C ISTRT - CHARACTER COUNT WHERE MOVE IS TO START FROM
9. C NOCHR - NUMBER OF CHARACTERS TO MOVE EACH TIME
10. C MOVCHR IS CALLED.
11. C
12. C DIMENSION FROM(11,TO(11)
13. C
14. C ISTRT = ISTRT
15. C RETURN
16. C
17. C
18. C
19. C ENTRY MOVCHR (TO)
20. C
21. C MOVCHR (MOVE CHARACTER) COPIES 'NOCHR' NUMBER OF CHARACTERS
22. C STARTING WITH 'ISTRT' CHAR IN FROM TO THE FIRST 'NOCHR'
23. C CHARACTERS IN 'TO'.
24. C EACH TIME MOVCHR IS CALLED AFTER MOVINT HAS BEEN CALLED, THE
25. C NEXT 'NOCHR'S ARE COPIED. MOVCHR KEEPS PROCESSING THE NEXT
26. C 'NOCHR'S UNTIL RE-INITIALIZED BY A CALL TO MOVINT.
27. C
28. C
29. C
30. C DO 30 J=1,NOCHR
31. C
32. C ISUB = ISTRT / 6 + 1
33. C ICHR = MOD(ISTRT,6)
34. C IF (ICHR.EQ.0) ISUB = ISUB - 1
35. C IF (ICHR.EQ.0) ICHR = 6
36. C IBT = ICHR + 6 - 6
37. C
38. C ISUB1 = J / 6 + 1
39. C IBT1 = MOD(J,6)
40. C IF (IBT1.EQ.0) ISUB1 = ISUB1 - 1
41. C IF (IBT1.EQ.0) IBT1 = 6
42. C IBT1 = IBT1 + 6 - 6
43. C
44. C FLD (IBT1,6,TO(ISUB1)) = FLD (IBT,6,FROM(ISUB))
45. C I = ISTRT + 1
46. C
47. C DO CONTINUE
48. C
49. C RETURN
50. C END

```

QELT,L N,NSCTRS,,NSCTRS

```
1:      FUNCTION NSCTRS (IARG)
2:      C
3:      ITMP = IARG / 28 + 1
4:      IF (MOD(28,IARG) .EQ. 0) ITMP = ITMP-1
5:      NSCTRS = ITMP
6:      RETURN
7:      END
```

WELT,L N,PKXCRU,PKXCRU

```

1* SUBROUTINE PKXCRU(DATA,NSEC)
2* IMPLICIT INTEGER(A-Z)
3*
4* DIMENSION DATA(1)
5*
6* DATA IS PACKED FROM CRUISE COMMON.
7*
8*
9* COMMON /CRUISE/CRUNO,PNATC,CATLAT,CATLON,CATDAT,CAFLAT,CAFLON,
10* CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
11* CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
12* CAFLG
13*
14* EQUIVALENCE (NOTES(1),NOTES(1,1))
15* DIMENSION NOTES(1)
16*
17* COMPUTE NUMBER OF WORDS IN NOTES AND CLEAN OUT ARRAY 'DATA'.
18*
19* N=14*NCARDS+17
20* DO 1 I=1,N
21* 1 DATA(I)=0
22*
23* FLD(16,1,DATA(1))=1
24*
25* COMPUTE AND PACK NUMBER OF SECTORS TO BE USED.
26*
27* NSEC=NSCTRS(N)
28* FLD(18,18,DATA(1))=NSEC
29*
30* PACK CRUISE NUMBER.
31*
32* FLD(0,18,DATA(2)) = CRUNO
33*
34* PACK NUMBER OF CARDS.
35*
36* FLD(18,18,DATA(2)) = NCARDS
37*
38* PACK POINTER TO NEXT CRUISE.
39*
40* DATA(3)=PNATC
41*
42* CALL PKAREA(' ',CATLAT,CATLON,CATDAT,0,' ',
43* CAFLAT,CAFLON,CAFDAT,0,' ',DATA(4))
44*
45* PACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
46*
47* CALL PKAREA(' ',CEILAT,CEILON,CEIDAT,0,' ',
48* CEFLAT,CEFLON,CEFDAT,0,' ',DATA(8))
49*
50* DATA(12)=CASTAT(1)
51* DATA(13)=CASTAT(2)
52*
53* DATA(14)=CESTAT(1)
54* DATA(15)=CESTAT(2)

```

```

55* C
56* DATA(14)=PACS
57* DATA(17)=PENS
58* C
59* C PACK NOTES*
60* C
61* N=14*NCARDS
62* DO 2 1=1,N
63* 2 DATA(1+17)=INOTES(1) MAY NEED EQUIV(INOTES(1),NOTES(1,1))
64* RETURN
65* END

```



AD-A038 891

NAVAL SEA SYSTEMS COMMAND WASHINGTON D C

F/G 17/1

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUP--ETC(U)

DEC 75

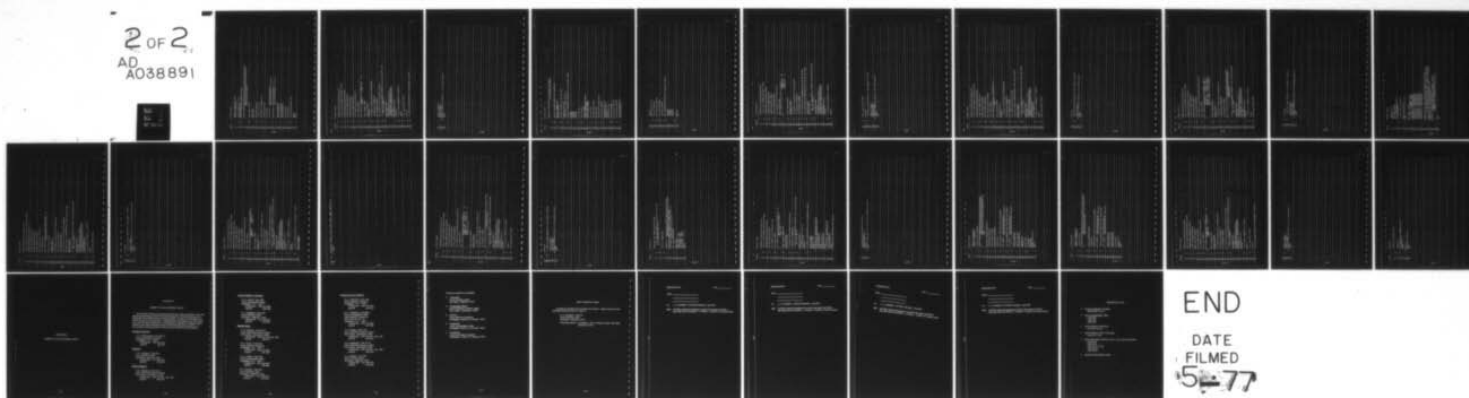
UNCLASSIFIED

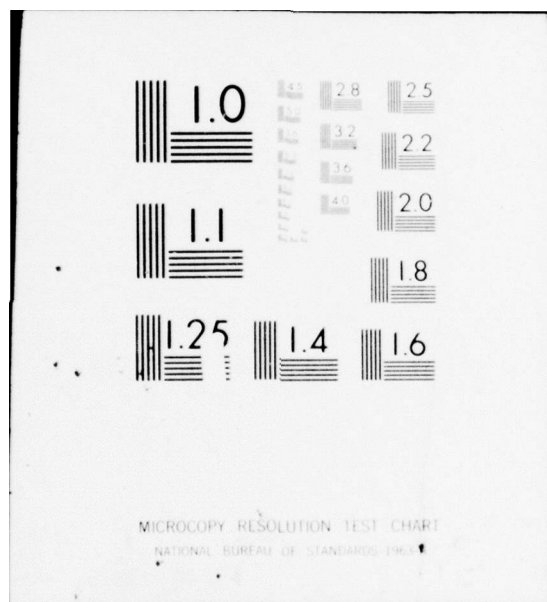
NAVSEA-06H1/036-EVA/MOST-

NL

2 OF 2

AD  
A038891





```

1. SUBROUTINE PCKEXP(DATA,NSEC)
2. IMPLICIT INTEGER(A-Z)
3.
4. C
5. C DATA IS PACKED INTO 'DATA'.
6. C DIMENSION DATA(1)
7. C DATA IS PACKED FROM EXPERIMENT COMMON.
8. C
9. C COMMON /EXPHT/EXNO,PNXTE,PCRU,EA1LAT,EA1LON,EA1DAT,EA1LAT,EA1LON,
10. C EAPDAT,EE1LAT,EE1LON,EE1DAT,EE1LAT,EE1LON,EE1DAT,
11. C EAPSTAT(2),EESTAT(2),EAPLG
12. C
13. C CLEAN ARRAY 'DATA'.
14. C
15. C DO I =1,20
16. C DATA(I)=0
17. C
18. C SET LEVEL TYPE BIT.
19. C
20. C FLD(17),DATA(1)=1
21. C
22. C SET NUMBER OF SECTORS FIELD.
23. C
24. C FLD(35),DATA(1)=1
25. C
26. C DATA(2)=EXNO
27. C
28. C CALL PKAREA(' ',EA1LAT,EA1LON,EA1DAT,0,' ',
29. C EAPLAT,EA1LON,EA1DAT,0,' ',DATA(3))
30. C
31. C CALL PKAREA(' ',EE1LAT,EE1LON,EE1DAT,0,' ',
32. C EE1LAT,EE1LON,EE1DAT,0,' ',DATA(7))
33. C
34. C DATA(11)=EASTAT(1)
35. C DATA(12)=EASTAT(2)
36. C
37. C DATA(13)=EEESTAT(1)
38. C DATA(14)=EEESTAT(2)
39. C
40. C DATA(15)=PNXTE
41. C
42. C DATA(16)=PCRU
43. C
44. C ONLY USED ONE SECTOR, 50...
45. C
46. C NSEC=1
47. C RETURN
48. C
49. C
50. C
51. C
52. C
53. C
54. C
55. C
56. C
57. C
58. C
59. C
60. C
61. C
62. C
63. C
64. C
65. C
66. C
67. C
68. C
69. C
70. C
71. C
72. C
73. C
74. C
75. C
76. C
77. C
78. C
79. C
80. C
81. C
82. C
83. C
84. C
85. C
86. C
87. C
88. C
89. C
90. C
91. C
92. C
93. C
94. C
95. C
96. C
97. C
98. C
99. C
100. C

```

OFOM,SA N,PERCNT,,PERCNT

1. SUBROUTINE PERCNT(VALUE,UNICODE,UNITS)  
2. C  
3. C TABLE OF PERCENT MEASURE CONVERSION FACTORS.

4. C  
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE

6. C  
7. C FACTORS ARE DOUBLE PRECISION.

8. C  
9. C DOUBLE PRECISION FACTOR(3)

10. C  
11. C PUT UNIT CODES INTO ENTRY.

12. C  
13. C DATA (ENTRY(1,1),1=1,3)/42,43,44/  
14. C  
15. C PUT IN ALPHA UNITS.

16. C  
17. C DATA ((ENTRY(1,1),1=2,3),J=1,3)/'PER C', 'NT',  
18. C ' (0/03)', ' ', 'PPH', ' ', ' ',  
19. C  
20. C ENTER INTERNAL UNITS.

21. C  
22. C UNITS(3)='PERCENT'  
23. C UNITS(4)='T'

24. C  
25. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.

26. C  
27. C IF (UNICODE.EQ.0) UNICODE=-42

28. C  
29. C SET CONVERSION FACTORS.

30. C  
31. C DATA (FACTOR(1,1),1=1,3)/1.000,0.100,1.00-4/  
32. C  
33. C IF CODE IS IN TABLE, PERFORM CONVERSION, ALSO, ENTER UNITS ALPHA  
34. C CODE INTO 'UNITS':  
35. C

36. C ICODE=ABS(UNICODE)  
37. C DO 2 J=1,3  
38. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2  
39. C IF (UNICODE.EQ.0) VALUE=VALUE\*FACTOR(1)  
40. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(1)  
41. C DO 1 J=1,2  
42. C UNITS(J)=ENTRY(J,1)  
43. C 1 CONTINUE

44. C  
45. C AT THIS POINT, CONVERSION IS COMPLETE.  
46. C  
47. C RETURN

48. C  
49. C ELSE, CHECK REST OF TABLE.  
50. C  
51. C 2 CONTINUE  
52. C  
53. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.  
54. C



55. UNITS(11) UNITS  
56. UNITS(12) ENORM  
57. PRINT 101, 1000  
58. 101 FORMAT( ENORM--UNIT CODE, 13, ' NOT IN PER CENT TABLE,')  
59. RETURN  
60. END

DELTY,L N,PKAREA,PKAREA

1. SUBROUTINE PKAREA(IPONA,ILAT,ILON,IOAT,ITIM,IZON,  
FLAT,FLOW,F0AT,FTIM,FZON,IOI)

2. C  
3. C  
4. C  
5. C  
6. C  
7. C  
8. C  
9. C  
10. C  
11. C  
12. C  
13. C  
14. C  
15. C  
16. C  
17. C  
18. C  
19. C  
20. C  
21. C  
22. C  
23. C  
24. C  
25. C  
26. C  
27. C  
28. C  
29. C  
30. C  
31. C  
32. C  
33. C  
34. C  
35. C  
36. C  
37. C  
38. C  
39. C  
40. C  
41. C  
42. C  
43. C  
44. C  
45. C  
46. C  
47. C  
48. C  
49. C  
50. C  
51. C  
52. C  
53. C  
54. C

MAKE ALL VARIABLES INTEGER AND DIMENSION IOT.

IMPLICIT INTEGER (A-Z)

DIMENSION IOT(4)

SET POINT OR AREA BIT (ON FOR AREA)

FLD(0,1,10T(1)) = 0

IF(IPONA.EQ.'A') FLD(0,1,10T(1)) = 1

INITIALIZE INTERMEDIATE VARIABLES TO INITIAL POINT VALUES.

IT1=IOAT

IT2=ITIM

IT3=IZON

IT4=ILAT

IT5=ILON

DO 4 I=1,12

PACK THE DATE.

ITMP=IT1/10000

ITMP=(IT1-ITMP\*10000)/100\*ITMP

IF(IT1.EQ.0) ITMP=0

FLD(1,17,10T(1))=ITMP

PACK THE TIME.

FLD(10,12,10T(1))=IT2

PACK THE ZONE.

FLD(30,5,10T(1))=FLD(1,5,IT3)

SET LATITUDE SIGN BIT

FLD(35,1,10T(1)) = 0

IF(IT4.GE.0) GOTO 2

IT4=-IT4

FLD(35,1,10T(1))=1

PACK THE LATITUDE.

2 FLD(0,17,10T(1))=IT4

SET LONGITUDE SIGN BIT.

FLD(17,1,10T(1)) = 0

IF(IT5.GE.0) GOTO 3

FLD(17,1,10T(1)) = 1

IF(IT5.GE.0) GOTO 3

```

55. ITS=ITS
56. FLD(17,1,10*(1+1))=1
57. C
58. C PACK THE LONGITUDE.
59. C
60. 3 FLD(10,10,10*(1+1))=ITS
61. C
62. C IF 1-3, WE ARE DONE.
63. C
64. C IF(1,9,3) RETURN
65. C
66. C SET INTERMEDIATE VARIABLES TO FINAL POINT VALUES.
67. C
68. IT1=FDAT
69. IT2=FTIM
70. IT3=FZON
71. IT4=FLAT
72. IT5=FLOW
73. C
74. C
75. 4 CONTINUE
76. END

```

—



```

55. C ELSE, CHECK REST OF TABLE.
56. C
57. C 2 CONTINUE
58. C
59. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
60. C
61. C UNITS(1)='UNITS '
62. C UNITS(2)='ERROR '
63. C PRINT 101,ICODE
64. C 101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN AREA TABLE.')
65. C RETURN
66. C END

```



QFOR,54

N,POWER,,POWER

```

1. SUBROUTINE POWOR(IVALUE,UNICDE,UNITS)
2. C
3. C TABLE OF POWER MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,2),UNITS(4),UNICDE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(2)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,2),21/59,60/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA (ENTRY(1,1),1,2),31,21/5963/5,1EC,
18. C
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='WATTS'
23. C
24. C
25. C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF (UNICDE.EQ.0) UNICDE=-60
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1,1),1,2),1.0D-7,100/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C
35. C UNITS(1)='UNITS'
36. C UNITS(2)='ERROR'
37. C
38. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
39. C CODE INTO 'UNITS'.
40. C
41. C ICODE=ABS(UNICDE)
42. C DO 2 1=1,2
43. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
44. C IF (UNICDE.EQ.0) VALUE=VALUE*FACTOR(1)
45. C IF (UNICDE.LT.0) VALUE=VALUE/FACTOR(1)
46. C UNITS(1)=ENTRY(2,1)
47. C UNITS(2)=ENTRY(3,1)
48. C
49. C AT THIS POINT, CONVERSION IS COMPLETE.
50. C
51. C RETURN
52. C
53. C ELSE, CHECK REST OF TABLE.
54. C

```

2 CONTINUE

55. C  
56. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.  
57. C  
58. C  
59. PRINT I01, ICODE  
60. I01 FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN POWER TABLE.')

RETURN  
END

N. PRESS. PRESS

1.	C	SUBROUTINE PRESS(IVALUE,UNICODE,UNITS)
2.	C	
3.	C	TABLE OF PRESSURE MEASURE CONVERSION FACTORS.
4.	C	
5.	C	INTEGER ENTRY(1,2),UNIT\$(14),UNICDE
6.	C	
7.	C	FACTORS ARE DOUBLE PRECISION.
8.	C	
9.	C	DOUBLE PRECISION FACTOR(19)
10.	C	
11.	C	PUT UNIT CODES INTO ENTRY.
12.	C	
13.	C	DATA (ENTRY(1,1),1,1,91/49,49,50,51,52,53,54,55,56/
14.	C	
15.	C	PUT IN ALPHA UNITS.
16.	C	
17.	C	DATA (ENTRY(1,2),1,2,3,4,5,6,7,MICRO,PASCAL,
18.	C	'PASCAL',5,'MICRO',PAS
19.	C	'KG/CM',1,0,0,2,LBS/LIN,0,2
20.	C	'LBS/F',1,0,0,2,'IN.ME',MCURY
21.	C	'MM.ME',BCURY,'ATMOSP',HERES
22.	C	
23.	C	ENTER INTERNAL UNITS.
24.	C	
25.	C	UNITS(3)=KG/CM'
26.	C	UNITS(4)=0,2
27.	C	
28.	C	IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
29.	C	
30.	C	IF(UNICODE.EQ.0) UNICDE=51
31.	C	
32.	C	SET CONVERSION FACTORS.
33.	C	
34.	C	DATA (FACTOR(1),1,1,91/101,97100,1,020-3,1,0200,100
35.	C	,4,9820-3,3,345362700,8,7722
36.	C	,1,03322288700/
37.	C	
38.	C	IF CODE IS IN TABLE, PERFORM CONVERSION, ALSO, ENTER
39.	C	CODE INTO UNITS.
40.	C	
41.	C	ICODE=IABS(UNICDE)
42.	C	DO 2 I=1,9
43.	C	IF(ICODE.NE.ENTRY(I,1)) GO TO 2
44.	C	IF(UNICODE.GE.0) VALUE=VALUE*FACTOR(I)
45.	C	IF(UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
46.	C	DO 1 J=1,2
47.	C	UNITS(J)=ENTRY(J+1,I)
48.	C	I CONTINUE
49.	C	
50.	C	AT THIS POINT, CONVERSION IS COMPLETE.
51.	C	
52.	C	RETURN
53.	C	
54.	C	ELSE, CHECK REST OF TABLE.

```

55. C
56. 2 CONTINUE
57. C
58. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
59. C
60. UNITS(1)='UNITS '
61. UNITS(2)='ERROR '
62. PRINT 101,ICODE
63. 101 FORMAT(' ERROR--UNIT CODE',13,' NOT IN PRESSURE TABLE.')
64. RETURN
65. END

```



GET,L N.PRINTO..PRINTJ

```

1. DIMENSION D(128)
2. INTEGER D
3. CALL NTRAN (10,22)
4. CALL SETADR(10,0)
5. CALL NTRAN(10,2,28,D,LSTAT)
6. CALL NTRAN(10,22)
7. C
8. IF (LSTAT.GT. 0) GO TO 20
9. C
10. WRITE (6,10) LSTAT
11. IQ FORMAT(' ERROR-- LSTAT =',15)
12. STOP
13. C
14. 20 IM = D(8) / 10003
15. ID = (D(8) - IM * 10003) / 109
16. IY = (D(8) - IM * 10003 - ID * 109) / 10
17. FLD(18,18,13) = FLD(10,18,0(10))
18. FLD(18,18,11) = FLD(18,18,0(10))
19. FLD(18,18,15) = FLD(18,18,0(14))
20. FLD(18,18,11) = FLD(18,18,0(14))
21. FLD(18,18,12) = FLD(18,18,0(15))
22. FLD(18,18,12) = FLD(18,18,0(15))
23. FLD(18,18,13) = FLD(18,18,0(14))
24. FLD(18,18,13) = FLD(18,18,0(14))
25. C
26. WRITE(6,25) (0(1),1(1),7),IM,10,1Y,D(9),15,1L,D(1),J(1),13)
27. 25 FORMAT (//////,1A,7A,/, ' DATE OF LAST UPDATE = ',12,/, '12,/, '12,
28. /, ' NEXT AVAILABLE SECTOR = ',110,/, ' ADDRESS OF LAST EXPER
29. IMENT LEVEL REC. = ',2110,/, ' LAST EXP. NO. = ',15,/, ' NO. CRUISES R
30. EMAINING = ',15,/, ' NO. STATIONS REMAINING = ',15)
31. WRITE (6,26) 15,11,11,12,13,11,3
32. 26 FORMAT (' ADDRESS OF LAST CRUISE LEVEL RECORD = ',2110,/,
33. ' ADDRESS OF LAST ACOUSTIC STATION = ',2110,/,
34. ' ADDRESS OF LAST ENVIRONMENTAL STATION = ',2110)
35. C
36. STOP
37. END

```

N. PRSLVL., PRSLVL

A-94

55. 2 CONTINUE

56. C

57. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.

58. C

59. PRINT I01, ICODE

60. IJ1 FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN PRESSURE LEVEL TABLE.')

61. RETURN

62. END

N. SHKIME, . SHKIME

SUBROUTINE SHRTME (VALUE, UNICDE, UNITS)

### TABLE OF SYNTHETIC MEASURE CONVERSION FACTORS.

INTEGER ENTRY(3,5),UNITS(4),UNICOE

**FACTORS ARE DOUBLE PRECISION.**

DOUBLE PRECISION FACTOR(S)

PUT UNIT CODES INTO ENTRY.

DATA (ENTRY(1,1),1-1,51/21,22,23,24,25/

**PUT IN ALPHA UNITS.**

DATA (ENTRY(1,J),1,2,3),J=1,51/'HILLIS', 'SECONDS',

```

..SECONDS ..
..HOURS ..
..DAYS ..
..MINUTES ..
.. /

```

ENTER INTERNAL UNITS.

UNITS(3) SEC. 1

UN113(4) = 0

MEMPHIS, TENN.

IF (UNICODE.EQ.0) UN

## SET CONVERSION FACT

DATA (FACTORY), 1980

IF CODE IS IN TABL  
CODE INTO 'UN

ICODE=195 (UNICODE)

00 2 1-1,5  
1x813-3M-3001131

IF (UNICODE,GE,0) VA  
IF (UNICODE,LT,0) VA

UNIVERSITY OF CALIFORNIA  
DO I J=1,2

CONTINUE

AT THIS POINT, COM

**RETURN  
CONTINUE**

IF CODE NOT FOUND

LIMITS OF -LIMITS !

UNIT(2) = ERROR '

1



56. 101 FORMATS, ERRORS--UNIT CODE, 12, NOT IN SHORT TIME TABLE, 1  
57. RETURN  
58. END

QFOR,58 N,SPEEDS,,SPEEDS

```

1: SUBROUTINE SPEEDS(VALUE,UNICODE,UNITS)
2: C
3: C TABLE OF SPEEDS MEASURE CONVERSION FACTORS.
4: C
5: C INTEGER ENTRY(3,0),UNITS(4),UNICODE
6: C
7: C FACTORS ARE DOUBLE PRECISION.
8: C
9: C DOUBLE PRECISION FACTOR(8)
10: C
11: C PUT UNIT CODES INTO ENTRY.
12: C
13: C DATA (ENTRY(1,1),1,0)/20,29,30,31,32,33,34,99/
14: C
15: C PUT IN ALPHA UNITS.
16: C
17: C DATA (ENTRY(1,1),1,2,3),J=1,8)/CM/SEC,,
18: C 'METERS',/SEC. ,/FT/SEC,,C.
19: C 'YARDS',/SEC. ,/KNOTS,,
20: C 'KM./HR.',/MILES/,,HR.
21: C 'FALMS',/FRTNT/
22: C
23: C ENTER INTERNAL UNITS.
24: C
25: C UNITS(3)=METERS
26: C UNITS(4)=/SEC.
27: C
28: C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
29: C
30: C IF UNICODE.EQ.0) UNICDE=-29
31: C
32: C SET CONVERSION FACTORS.
33: C
34: C DATA (FACTOR(1),1,0,1,8)/0.100,100,,304800,,914400,,514500,,27777700
35: C
36: C
37: C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
38: C CODE INTO UNITS.
39: C
40: C ICODE=ABS(UNICODE)
41: C DO 2 I=1,8
42: C IF ICODE.NE.ENTRY(I,1) GO TO 2
43: C IF ICODE.GE.0) VALUE=VALUE*FACTOR(I)
44: C IF ICODE.LT.0) VALUE=VALUE/FACTOR(I)
45: C DO I J=1,2
46: C UNITS(J)=ENTRY(J,1)
47: C I CONTINUE
48: C
49: C AT THIS POINT, CONVERSION IS COMPLETE.
50: C
51: C RETURN
52: C
53: C ELSE, CHECK NEST OF TABLE.
54: C

```

2 CONTINUE

55. C  
56. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR IN 'UNITS'.  
57. C  
58. C  
59. UNITS(1)='UNITS '  
60. UNITS(2)='ERROR '  
61. PRINT 101,1CODE  
62. 101 FORMAT(' ERROR--UNIT CODE',13,' NOT IN VELOCITY TABLE.')

RETURN  
END

9FOR,34 N,STOR,STOR

```

1. SUBROUTINE STOR (CODE,VALUE,UNIT,METHOD)
2. C
3. C STOR (STORE) MOVES PARAMETERS INPUT ON THE ENII CARD
4. C TO THE CONSTANTS AREA OF /CPV/.
5. C
6. C
7. C IMPLICIT INTEGER (A-Z)
8. C
9. C
10. COMMON /CPV /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
11. * NDATS,NNDATS(50),NDATA,VALRNG(2,30,53),DATA(30000)
12. * DIMENSION ICON(4,30),IPAR(30,30),IVLNG(2,30,60)
13. * EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
14. * (VALRNG(1,1),IVLNG(1,1))
15. * REAL CON,PAR,DATA,VALRNG
16. C
17. REAL VALUE
18. NCON = NCON + 1
19. ICON(1,NCON) = CODE
20. ICON(2,NCON) = UNIT
21. CON (3,NCON) = VALUE
22. ICON(4,NCON) = METHOD
23. RETURN
24. END

```



9FOR,54 N,IMPTUR,,IMPTUR

1. SUBROUTINE IMPTUR(VALUE,UNICODE,UNITS)  
2. C  
3. C TABLE OF IMPTUR MEASURE CONVERSION FACTORS.  
4. C  
5. C INTEGER ENTRY(3,4),UNITS(1),UNICODE

6. C  
7. C PUT UNIT CODES INTO ENTRY.  
8. C  
9. C DATA ENTRY(1,1),1=1,1)/35,36,97,98/  
10. C  
11. C PUT IN ALPHA UNITS.  
12. C  
13. C DATA (ENTRY(1,1),1=2,3),1=1,1)/FARRENH,MEIT ,  
14. C ,DEG, C, CELSIUS, KELVIN, ,  
15. C ,RANKIN, 'ES ,/  
16. C  
17. C ENTER INTERNAL DEGREE UNITS.  
18. C  
19. C UNITS(1)=DEGREE  
20. C UNITS(4)=5 CENT  
21. C  
22. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.  
23. C  
24. C IF(UNICODE.EQ.0) UNICODE=-36  
25. C  
26. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA  
27. C CODE INTO 'UNITS'.  
28. C  
29. C ICODE=ABS(UNICODE)  
30. C DO 1 1=1,4  
31. C IF(ICODE.NE.ENTRY(1,1)) GO TO 1  
32. C GO TO (2,3,4,5),1  
33. C 1 CONTINUE  
34. C  
35. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.  
36. C  
37. C UNITS(1)='UNITS '  
38. C UNITS(2)='ERROR '  
39. C PRINT 101,ICODE  
40. C RETURN  
41. C 2 IF(UNICODE.GE.0) VALUE=(VALUE-32.)\*.5./9.  
42. C IF(UNICODE.LT.0) VALUE=VALUE\*.75+.32.  
43. C UNITS(1)=ENTRY(2,1)  
44. C UNITS(2)=ENTRY(3,1)  
45. C RETURN  
46. C 3 UNITS(1)=ENTRY(2,1)  
47. C UNITS(2)=ENTRY(3,1)  
48. C RETURN  
49. C 4 IF(UNICODE.GE.0) VALUE=VALUE-273.15  
50. C IF(UNICODE.LT.0) VALUE=VALUE+273.15  
51. C UNITS(1)=ENTRY(2,1)  
52. C UNITS(2)=ENTRY(3,1)  
53. C RETURN  
54. C 5 IF(UNICODE.GE.0) VALUE=(VALUE-491.67).5./9.

```
55. IF (UNICDE.LT.0) VALUE=VALUE*9./5.+491.67
56. UNITS(1)=ENTRY(2,1)
57. UNITS(2)=ENTRY(3,1)
58. RETURN
59. 101 FORMAT(10 ERRUR--UNIT CODE ',13,' NOT IN TEMPERATURE TABLE.')
```

```
END
```

0817, L N, UNPACK, UNPACK

```

1. SUBROUTINE UNPACK(DATA)
2. IMPLICIT INTEGER(A-Z)
3. C
4. C DIMENSION DATA(1), NOTES(1)
5. C
6. COMMON /CRUISE/CRUNO, PHATC, CATLAY, CAILOM, CAIDAT, CAPLAY, CAPLON,
7. CAPDAT, CEILAT, CEILOM, CEIDAT, CEFLAT, CEFLON, CEFOAT,
8. C, CASTAT(2), CESTAT(2), PACS, PENS, NCARDS, NOTES(1, 132),
9. CAPLG
10. C
11. EQUIVALENCE(1, NOTES(1), NOTES(1, 1))
12. C
13. CRUNO=FLD(10, 18, DATA(2))
14. C
15. C UNPACK NUMBER OF CARDS.
16. C
17. NCARDS=FLD(10, 18, DATA(2))
18. C
19. PHATC=DATA(3)
20. C
21. C UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
22. C
23. CALL UPREA(' ', CATLAY, CAILOM, CAIDAT, 0.0, ' ',
24. CAPLAY, CAPLON, CAPDAT, 0.0, DATA(4))
25. C
26. C UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
27. C
28. CALL UPREA(' ', CEILAT, CEILOM, CEIDAT, 0.0, ' ',
29. CEFLAT, CEFLON, CEFOAT, 0.0, DATA(8))
30. C
31. CASTAT(1)=DATA(12)
32. CASTAT(2)=DATA(13)
33. C
34. CESTAT(1)=DATA(14)
35. CESTAT(2)=DATA(15)
36. C
37. PACS=DATA(16)
38. PENS=DATA(17)
39. C
40. C UNPACK NOTES.
41. C
42. N=14*NCARDS
43. DO 1 I=1, N
44. I NOTES(1)=0
45. I NOTES(1)=DATA(1, 17)
46. RETURN
47. END

```

QELT,L N,UNPKXP,UNPKXP

```

1. SUBROUTINE UNPKXP(DATA)
2. IMPLICIT INTEGER(A-Z)
3. C
4. C DIMENSION DATA(1)
5. C
6. COMMON /EXPNT/EXNO,PNTX,PCRU,EALAT,EALON,EALDAT,EAFAT,EAFON,
7. EAFDAT,EALAT,EALON,EALDAT,EAFAT,EAFON,EAFDAT,
8. EAFAT,EALAT,EALON,EALDAT,EAFAT,EAFON,EAFDAT,
9. EAFAT,EALAT,EALON,EALDAT,EAFAT,EAFON,EAFDAT
10. C
11. C UNPACK EXPERIMENT NUMBER.
12. C
13. C EXNO=DATA(2)
14. C
15. C UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
16. C
17. CALL UPKREAL('EALAT,EALON,EALDAT',
18. EAFAT,EAFON,EAFDAT,0.0,DATA(3))
19. C
20. C UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
21. C
22. CALL UPKREAL('EALAT,EALON,EALDAT',
23. EAFAT,EAFON,EAFDAT,0.0,DATA(7))
24. C
25. EAFAT(1)=DATA(11)
26. EAFAT(2)=DATA(12)
27. C
28. EAFAT(1)=DATA(13)
29. EAFAT(2)=DATA(14)
30. C
31. PNTX=DATA(15)
32. C
33. PCRU=DATA(16)
34. RETURN
35. END

```



SUBROUTINE VOLUME(VALUE,UNIQUE,UNITS)  
TABLE OF VOLUME MEASURE CONVERSION FACTORS.

### TABLE OF VOLUME MEASURE CONVERSION FACTORS:

INTEGER ENTRY(3,4),UNITS(4),UNICDE

**FACTORS ARE DOUBLE PRECISION.**

**DOUBLE PRECISION FACTOR(4)**

PUT UNIT CODES INTO ENTRY.

DATA (ENTRY(1,1),10,1,4)/17,10,19,20/

PUT IN ALPHA UNIT.

DATA (ENTRY(1,1),I=2,3),J=1,4)/CMO3 0.0 0.0  
0MYENS0.0003 0.0FEET000.03 0.0

**FOOTNOTES**

**ENTER INTERNAL UNITS:**

UNITS(3) = METERS.  
UNITS(4) = CM.

IS UNICOF-0. The

הוא מנסה להבין את המצב, ואת הסיבות, ואת הפתרונות.

01203031NO (0.03'3031NO) 41

## SEE CONVERSION FACTORS

DATA: (FACTOR(1),101,4)/1.00-6,100,2.03170-2,0.7640007

IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA  
CODE INTO 'UNITS'.

(3001MN)50V(030021)

00 2 1=1,4  
11461M3-3M-300J1131

IF (CODE,NE

IF (UNIQUE.GE.0) VALUE=VALUE/FAC;  
IF (UNIQUE.LT.0) VALUE=VALUE/FAC;

```
IF(UNIQUE(T.O) VALUE=VALUE/FACTOR(I))
```

```
DO 1 J=1,2
UNIT5(J)=ENTRY(J+1,1)
```

**CONTENTS**

AT THIS POINT, CONVERSATION

AT 1113 P

RETURN

0 3573

0.17701 10 1524 47673

**CONTINUE**

IF CODE N

1. The first step in the process of identifying a problem is to recognize that a problem exists. This involves gathering information about the situation and identifying the specific issue that needs to be addressed.

55.  
56.  
57.  
58.  
59.  
60.  
61.

C  
UNITS(11)=UNITS  
UNITS(12)=ERROR  
PRINT 101,ICODE  
101 FORMAT('ERROR--UNIT CODE',I3,' NOT IN VOLUME TABLE.')

RETURN  
END

0817.1 N.WEST..WEST

```
1:      INTEGER FUNCTION WEST (A,B)
2:      C
3:      C
4:      C
5:      C
6:      IF (ABS(A-B) .GT. 100000) GO TO 10
7:      WEST = MIN(A,B)
8:      RETURN
9:      C
10:     10 WEST = B
11:     IF (A .GT. 0) WEST = A
12:     RETURN
13:     END
```

APPENDIX B  
CURRENT NAVDAB STEERING GROUP



## APPENDIX B

### CURRENT NAVDAB STEERING GROUP

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is under the general sponsorship and direction of NAVSEA 06H1-4. NAVDAB functions as a subsidiary of the Mobile Sonar Technology (MOST) Acoustics of the Medium Committee (ACME) for NAVSEA 06H1. Dissemination of information and requests for any of the documents pertaining to NAVDAB may be obtained by sending a formal request to any of the following current NAVDAB Steering Group members. A list of official mailing addresses follows the list of Steering Group members.

#### Executive Secretary

A. P. Franceschetti, SEA 06H1-4  
Naval Sea Systems Command  
Washington, D.C. 20362  
Commercial (202) 692-3166  
Autovon 222-3166

#### Chairman

L. A. Smothers, Code 3073  
Naval Undersea Center  
San Diego, California 92132  
Commercial (714) 225-2316  
Autovon 933-2316

#### Branch Managers

F. R. DiNapoli, Code TA113  
Naval Underwater Systems Center  
New London, Connecticut 06320  
Commercial (203) 442-0771 Ext. 2647  
Autovon 636-2647

Branch Managers (Continued)

W. H. Geddes, Code 3440  
Naval Oceanographic Office  
Washington, D.C. 20373  
Commercial (202) 433-3994  
Autovon 288-3994

L. A. Smothers, Code 3073  
Naval Undersea Center  
San Diego, California 92132  
Commercial (714) 225-2316  
Autovon 933-2316

Steering Group

F. R. DiNapoli, Code TA113  
Naval Underwater Systems Center  
New London, Connecticut 06320  
Commercial (203) 442-0771 Ext. 2647  
Autovon 636-2647

R. H. Ferris, Code 8120  
Naval Research Laboratory  
Washington, D.C. 20375  
Commercial (202) 767-3359  
Autovon 297-3359

W. H. Geddes, Code 3440  
Naval Oceanographic Office  
Washington, D.C. 20373  
Commercial (202) 433-3994  
Autovon 288-3994

R. F. Hosmer, Code 3071  
Naval Undersea Center  
San Diego, California 92132  
Commercial (714) 225-2316  
Autovon 933-2316

Steering Group (Continued)

W. A. Kuperman, Code 8120  
Naval Research Laboratory  
Washington, D.C. 20375  
Commercial (202) 767-3210  
Autovon 297-3210

K. V. Mackenzie, Code 9420  
Naval Oceanographic Office  
Naval Research Laboratory  
4555 Overlook Ave. SW  
Washington, D.C. 20375  
Commercial (202) 767-2830  
Autovon 297-2830

L. C. Maples, Code TA1A  
Naval Underwater Systems Center  
New London, Connecticut 06320  
Commercial (203) 442-0771 Ext. 2501  
Autovon 636-2501

S. R. Santaniello, Code TA111  
Naval Underwater Systems Center  
New London, Connecticut 06320  
Commercial (203) 442-0771 Ext. 2675  
Autovon 636-2675

J. A. Whitney, Code 3073  
Naval Undersea Center  
San Diego, California 92132  
Commercial (714) 225-2316  
Autovon 933-2316

#### OFFICIAL MAILING ADDRESSES

1. Commander  
Naval Undersea Center  
San Diego, California 92132
2. Commanding Officer  
Naval Underwater Systems Center  
New London, Connecticut 06320
3. Director  
Naval Research Laboratory  
Washington, District of Columbia 20375
4. Commander  
Naval Oceanographic Office  
Washington, District of Columbia 20373
5. Commander  
Naval Sea Systems Command  
Washington, District of Columbia 20362



# USER COMMENTS FORM

Comments concerning this document are invited. Please send one of the following attached forms or a letter to:

L. A. Smothers, Code 307  
Naval Undersea Center  
San Diego, California 92132

(Envelope address: Commander, Naval Undersea Center, San Diego,  
California 92132)

MEMORANDUM

Date: \_\_\_\_\_

From: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),  
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)

CUT

MEMORANDUM

Date: \_\_\_\_\_

From: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),  
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)

CUT

MEMORANDUM

Date: \_\_\_\_\_

From: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),  
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)



MEMORANDUM

Date: \_\_\_\_\_

From: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),  
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)

CUT

## DISTRIBUTION LIST

- 5    Naval Sea Systems Command  
      NSEA-06H1 (5 cys)
  
- 3    Naval Oceanographic Office  
      Code 3432  
      Code 3440  
      Code 6100
  
- 2    Naval Research Laboratory  
      Code 8120 (2 cys)
  
- 7    Naval Undersea Center, San Diego  
      Code 307 (7 cys)
  
- 6    Naval Underwater Systems Center, New London Laboratory  
      Code TA1A  
      Code TA42  
      Code TA111 (2 cys)  
      Code TA112  
      Code TA113
  
- 2    Defense Documentation Center